Criminal Data Analysis Based on Low Rank Sparse Representation

Turki Fisal Aljrees

Faculty of Science and Technology

Middlesex University

A thesis submitted to Middlesex University in partial fulfilment of the requirments for the degree of

Doctor of Philosophy

June 2019

For my wife,
Edyta Pozoga.

I would like as well to dedicate this thesis to my loving kids, and parents.
And to ALMIGHTY GOD who always blesses us!

Declaration

This thesis is an account of research undertaken at The faculty of Science and Technology, Middlesex University London, United Kingdom. Except where acknowledged in the customary manner, the material presented in this thesis is, to the best of my knowledge, original and has not been submitted in whole or part for a degree in any university. This dissertation contains more than 34,982 words excluding appendices, bibliography, footnotes, tables and equations and has more than 80 figures.

Turki Fisal Aljrees

June 2019

Acknowledgements

I would like first to acknowledge my thanks to all my PhD advisers Dr. Daming Shi, Dr. David Windridge, and Dr Xiaochun Cheng for supporting me during these past years. All my supervisors are instantly respectful and never forgettable. Daming has enhanced my interest in Machine Learning and pushed my technical boundaries. Through Daming I developed my critical thinking skills and was encouraged to establish the quality of work expected. While the feedback provided by Daming helped to structure my research for an academic audience and to develop a high level researcher across the different stages along the PhD journey. David has been supportive and has given me the freedom to various methods without objection. His valuable advice that provided me insightful discussions about the research. Xiaochun has been supportive and has given me the full access to his time and knowledge with complete trust. Xiaochun has also provided me very good feedback, and I am very happy to know him and work with him. I hope that I could be as enthusiastic, and energetic as Xiaochun and to someday be able to command an audience as well as he can. I also thank the members of my PhD committee, Professors, and staff for their helpful advice and suggestions in general. I especially thank my mum, brothers, and sister, for their relationship and encouragement. I dedicate this thesis to my family, my wife, Edyta, and my three beautiful kids, Sara, Maria, and Newborn Joseph for their unconditional love, patient, and constant support.

Abstract

FINDING effective clustering methods for a high dimensional dataset is challenging due to the curse of dimensionality. These challenges can usually make the most of basic common algorithms fail in high-dimensional spaces from tackling problems such as large number of groups, and overlapping. Most domains uses some parameters to describe the appearance, geometry and dynamics of a scene. This has motivated the implementation of several techniques of a high-dimensional data for finding a low-dimensional space. Many proposed methods fail to overcome the challenges, especially when the data input is high-dimensional, and the clusters have a complex.

REGULARLY in high dimensional data, lots of the data dimensions are not related and might hide the existing clusters in noisy data. High-dimensional data often reside on some low dimensional subspaces. The problem of subspace clustering algorithms is to uncover the type of relationship of an objects from one dimension that are related in different subsets of another dimensions. The state-of-the-art methods for subspace segmentation which included the Low Rank Representation (LRR) and Sparse Representation (SR). The former seeks the global lowest-rank representation but restrictively assumes the independence among subspaces, whereas the latter seeks the clustering of disjoint or overlapped subspaces through locality measure, which, however, causes failure in the case of large noise.

THIS thesis aims are to identify the key problems and obstacles that have challenged the researchers in recent years in clustering high dimensional data, then to implement an effective subspace clustering methods for solving high dimensional crimes domains for both real events and synthetic data which has complex data structure with 168 different offence crimes. As well as to overcome the disadvantages of existed subspace algorithms techniques. To this end, a Low-Rank Sparse Representation (LRSR) theory, the future will refer to as Criminal Data Analysis Based on LRSR will be examined, then to be used to recover and segment embedding subspaces. The results of these methods will be discussed and compared with what already have been examined on previous approaches such as K-mean and PCA segmented based on K-means. The previous approaches have helped us to chose the right subspace clustering methods. The Proposed method based on subspace segmentation method named Low Rank subspace Sparse Representation (LRSR) which not only recovers the low-rank subspaces but also gets a relatively sparse segmentation with respect to disjoint subspaces or even overlapping subspaces.

BOTH UCI Machine Learning Repository, and crime database are the best to find and compare the best subspace clustering algorithm that fit for high dimensional space data. We used many Open-Source Machine Learning Frameworks and Tools for both employ our machine learning tasks and methods including preparing, transform-

ing, clustering and visualizing the high-dimensional crime dataset, we precisely have used the most modern and powerful Machine Learning Frameworks data science that known as **SciKit-Learn** for library for the Python programming language, as well as we have used R, and Matlab in previous experiment.

Abbreviations

Acronyms / Abbreviations

AI - Artificial intelligence

ML - Machine Learning

KDD - Knowledge Discovery from Data

kNN - k nearest neighbor

PCA - Principal Component Analysis

SVD - Singular Value Composition

RPCA - Robust Principal Component Analysis

FS - Feature Selection

PCR - Principal Component Regression

PLSR - Partial Least Squares Regression

GPCA - Generalized Principal Component Analysis

MDS - Multidimensional Scaling

LDA - Linear Discriminant Analysis

LLE - Locally Linear Embedding

LSA - Local Subspace Affinity

LLMC - Locally Linear Manifold Clustering

EM - Expectation-Maximization Algorithm

SS - Subspace Segmentation

SC - Spectral Clustering

SR - Sparse Representation

LSC - Local Spectral Clustering

SSC - Sparse Subspace Clustering

LRR - Low Rank Representation

LRSR - Low Rank Subspace Sparse Representation

KKT - Karush–Kuhn–Tucker conditions

Contents

List of Figures

List of Tables

List of Publications

[1] TURKI ALJREES, DAMING SHI, DAVID WINDRIDGE, WILLIAM WONG '**CRIMINAL PATTERN IDENTIFICATION BASED ON MODIFIED K-MEANS CLUSTERING**', ©2016 IEEE, Proceedings of the 2016 International Conference on Machine Learning and Cybernetics, Jeju, South Korea, 10-13 July, 2016

[2] TURKI ALJREES, DAMING SHI, DAVID WINDRIDGE, XiaochunIAOCHUN CHENG '**CRIMINAL DATA ANALYSIS BASED ON SUBSPACE CLUSTERING**',(Draft) October, 2018

[3] TURKI ALJREES, DAMING SHI, DAVID WINDRIDGE, XiaochunIAOCHUN CHENG '**LRSR SUBSPACE TO CRIMINAL DATA ANALYSIS**',(Draft) 2018

Chapter 1

Introduction

Man is the best computer we can put aboard a spacecraft, and

the only one that can be mass produced with unskilled labor

(Wernher von Braun) .

1.1 Background

Whenever we look around, there always activities which have been monitored and recorded, for example people movements on CCTV, communication by phones, emails or even by the social media. As a result, high-dimensional data has been created and need to be interpreted into meaningful content, consequently, advance techniques on how to learn, compress, store, transmit, and process the data are highly needed.

AI research always focus on tasks that easier for machine and even simple applications such as rapid calculations, playing chess, automated theorem proving are easily to be master by computer, but historically hard for humans. The advancement of Artificial Intelligence (AI) failed over past 60 years, and the main goal to make machines understand the human language, still Is too early.

The gap between the amount of digitally available data and human ability to analyse and understand is becoming wider. Numerous methods that offer excellent results need too many resources to be applied to modern datasets. Problems that were considered “solved” decades ago re-appear, as increasing the amount of input data exponentially opens up new challenges, require new methods to solve. There are always more challenging problems, a trickier dataset,

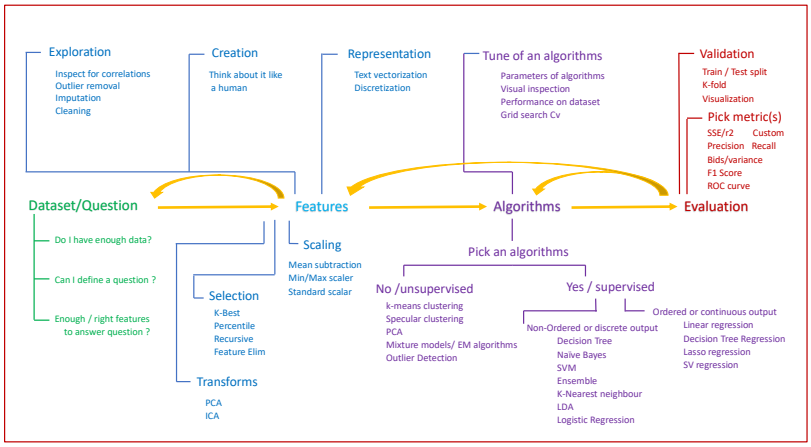


Figure 1.1 Four main Machine Learning blocks techniques

another group of dissatisfied users. The most exciting Machine Learning problems always seem to be one step ahead of any available, proven “solutions”.

(ML) Machine Learning is concerned with discovering useful patterns in large datasets, often for making decisions or predictions. a large set of statistical and programmatic techniques (i.e., clustering, etc.). Machine learning employs both clustering data and statistical techniques with the explicit goal of enabling machines to understand a set of data. Techniques may be supervised or unsupervised. Often these techniques are used in conjunction with statistical methods for elucidating complex relationships such as non-linear [10].

In the field of ML and AI, there are many techniques that are used to extract features and inherent structures from a high dimensional data such as image or sequence of images, to cluster the data into groups or classes. Extracted features, i.e. straight-lines and segments, or clustering, i.e. Subspace Clustering (SC).

The current state-of-the-art Machine Learning (ML) techniques that contain many mathematical models often hard to understand and tough for human experts to translate. In addition to the challenges, their output not often translates straight forward to different or new information about the problem. Powerful ML Algorithm needed to summarise not only single data but high dimension space to be described as less information as possible.

In ML, there are four main different blocks of techniques: Data, Features, Algorithm, and Evaluation as in Figure 1.1. The Data set or the questions that determine must always inspire us, and then extracting the data in the form of features, that feed these features to the machine learning algorithm as hart of the clustering and machine learning techniques. Finally, the

evaluation process that is used validate the performance of algorithm. we can sum them up into the following 1.1.

The goals of the prediction and description are achieved by using the following primary data mining tasks; classification, or clustering. The classification task is learning a function that maps (classifies) a data item into one of several predefined classes. Regression is a to data item to a real-valued prediction variable, while the clustering is a common descriptive task where one seeks to identify the group inherent in the data.

K-means The purpose of clustering, in general, is to group the data based on the similarities or dissimilarities. There are many famous clustering algorithms for example, k-means cluster, two-step cluster, and hierarchical clustering. For applications in which a number of clusters are required, K-means is very useful; similarly, the k-means clustering technique groups based on similarity

Modified K-means There are some obstacle and limitations when applying K-means on the large dataset, as described in [11]. Modified K-means algorithm was proposed to over come the K-means issues an designed to reduce the complexity algorithm. The algorithm was constructed based on the density of the different regions, and the idea was that object that displayed the minimum distance for a data point became the cluster for that data point. Modified k-means features an alternative way of calculating and computing the frequency of each data point in segments and dividing the entire space into more than a few segments. The partition of the space is used to calculate the frequency of the input vector in every part and to pick the highest k-frequency section. The modified k-means approach is used to optimise the current k-means algorithm. The K-means limitation resolved by running the algorithm for different numbers of k values but still fail with big data and high dimensional space.

1.1.1 Criminal Data Analysis

Criminal Data Analyst recently have increased the use of Machine Learning algorithms such as clustering as an effective method. Many of these methods have been used to find patterns or for prediction purpose, but still there is great need in developing perfect model and most of these methods must take into account many factors such as labelling the availability data, calculating Times, and counting frequents[12].

The crime patterns are always changing and growing [13]. The crime data that stored from several sources have a tendency to increase steady. To solve the problems previously

mentioned, data mining method apply numerous learning algorithms to extract hidden knowledge from huge volume of data. Data mining techniques for finding patterns and trends in crimes. It can assist in solving crimes much quickly and can assist to warn for potential criminal activities . Various data mining method are exploited for other goal such as criminality, science, finance, banking, email filtering, healthcare and other industries. nonetheless [14].

Criminal analysis is analytical process that provides useful information to assist in the planning to prevent growing of criminal activities, and it also involves in many tasks for identifyng patterns to help in a more effective manner. In criminal analysis, it can be much harder to grasp the motivations for patterns when the pattern changed to create a better plan for the future occasions, as it includes several factors.

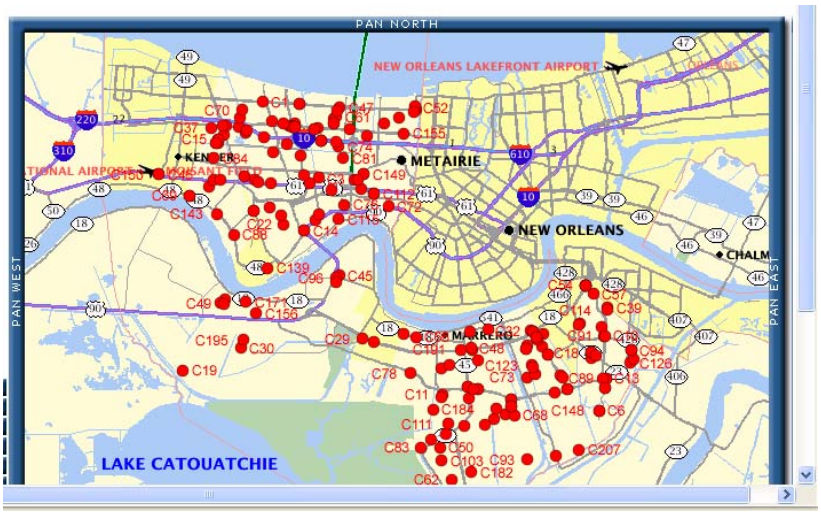


Figure 1.2 Geo-spatial plot of crimes, each red dot represents a crime incident.[1]

The following following direction been study as the main crime types that been the interest of scientist and investigator: Traffic Violation and Border Control, Violent Crime, The Narcotics , and Cyber Crime [15]

In social science, criminal analysis is a cluster analysis which used to identify areas where there are greater incidences of particular types of crime. By identifying these distinct areas or "hot spots" [1]

[1] "hot spots" The densely populated group of crime is used to visually locate the 'hot-spots' of crime such as fig. 1.2. However, when we speak of clustering from a data-mining standpoint, we refer to similar kinds of crime in the given geography of interest. [1]

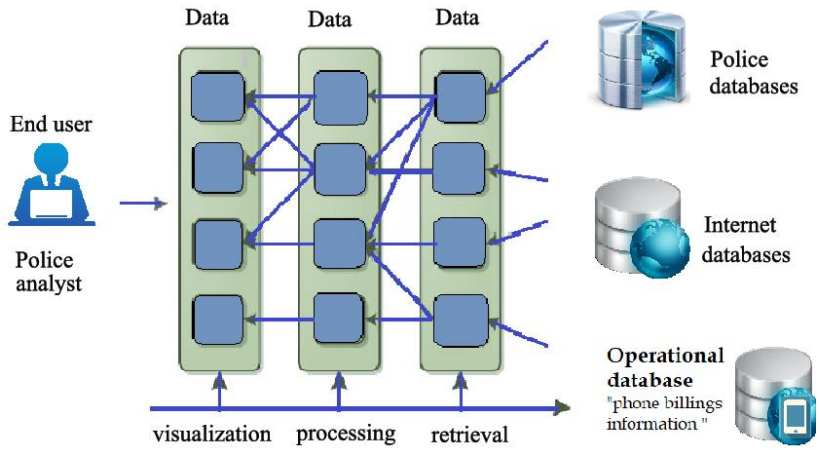


Figure 1.3 Criminal Analysis Process Loop

where a similar crime has happened over a period of time, it is possible to manage law enforcement resources more effectively like assisting to identify suspects.

The study of Data mining in the criminology analysis can be considered into areas, like crime pattern detection and crime control. De Bruin et. al. [16] introduced a framework for crime trends using a new distance measure for all individuals based on their profiles, then cluster them accordingly. Still, the outcomes must be carefully analysed to guarantee that are important. This regularly requires heuristic methodologies, which again make few assumptions about the data and or the clusters to be found. This study about the crime pattern analysis helps the detectives, but not replace them.

Criminal Data Process Issues

Criminal analysis tasks often evolve as a loop of intertwined process: Data Collection and Integration, data-retrieval, processing, visualisation, Crime Pattern, Performance, and interpretation. The aims and rule depends on a given project as given in fig. 1.3. various information used by different type of analyst operators such as police and Internet services and operational databases [17]. The databases retrieval can be from a particular source and it requires a specific tools, then database can be searched, filtered, or analysed with the use of algorithms such as data mining , pattern recognition algorithms [18] or hypothesis testing based on social networks approach [19], [20]. Frequently, there is a specific range of operations which can be executed on a data coming from a given source. The result of the data processing stage is usually a subset of the initial data, that can be seen in big picture by using visualization types [17].

A major issue crime analysts confront is why their statistics do not match the “official” statistics reported to the FBI. From the discussion above, we can see how this happens; any of the following could be true: The crime analyst downloaded the data at a different time than the Uniform Crime Report (UCR) or National Incident Based Reporting System (NIBRS) data were downloaded . The crime analyst uses state crime codes, not UCR or NIBRS codes. The crime analyst uses only those crimes determined by the police (not the victim) to be a crime.

The crime analyst counts by number of reports, not the number of victims or by the hierarchy of crimes. All of these types of data serve a purpose in providing a picture of crime [21]. Numerous studies have shown that crime data analysis is sensitive in nature and require domain knowledge before applying any dimensionality reduction techniques as it may result low level of accuracy. The problems of interacting with large datasets in Intelligence-Analysis require analysts to process large amounts of data into actionable intelligence, i.e. information that can be used to initiate investigations or police actions that may be able to prove innocence or guilt.

Such data can be structured or un-structured, quantitative and qualitative, of multiple formats, e.g. text, documents, images, videos, or streaming data such as social media or news feeds; be from multiple sources, be of varying quality and reliability, sparse, streaming, and represent rapidly changing situations. It is no longer humanly possible to sift through all the necessary data. Systems are needed to support this task. This presents many difficulties and challenges to the intelligence analysts. Given any set of data, there are endless combinations of attributes to represent the data. This study [22] shows guidance on best practice for using crime and policing statistics to improve understanding and interpretation of the data and to help build and maintain public trust in official information, and to follow the framework which is based on three principle: Trustworthiness, Quality and Value. Together, these pillars support public confidence in statistics. For this work we will focused more on the Quality which is about the data, and how they are processed into applied mathematics. Following the Review of Economic Statistics report [23] , the Code recognises that independence of production is not, on its own, enough to guarantee worthwhile statistics. The statistics must be the best available estimate of what they aim to measure, and should not mislead.

To achieve this, the data must be relevant, the methods must be sound and the assurance about the outputs must be clear. These aspects of statistical production are at the heart of the practices in the Quality pillar [22].

To conclude, the changing and increasing of crime lead to the issues of understanding the crime behaviour, predicting, detection, and managing of the big data . Research interests have attempt to resolve these issues. yet, these still gaps in the crime detection accuracy. This extend the challenges even more of crime detection. The challenges include modelling of suitable algorithms, data preparation and transformation, and processing [15] .

1.1.2 Big Data Analysis

The age of big data has arrived. Merely the existed data analytics tools is not be able to deal with large quantity of data. How to develop an algorithm or a high performance system to better analyse the big data and to find the useful pattern in big dataset is demanding and becoming researcher focuses. [24]

The Properties Of Big Data

Yet is very difficult for analyst to extract information in big data because it requires large parallel processing since the data is heterogeneous. But what is the difference between the term heterogeneous and homogeneous? In **Computer Science** : $HeTero = Variables$ of different nature in the same data set. $Categorical + binary + ...$ where $HoMo = Variables$ of same nature in the same data set. $Only binary, ...$ on the other hand In **Data Clustering**: it also described that $HoMogeneous$: i.e data that belong to the same **cluster** should be as similar as possible. where $HeTerogeneous$: i.e data that belong to different clusters should be as different as possible. In General definitions: $HeTero = a$ combining form meaning "Different" and “Other” , where $HoMo = a$ combining form meaning “Same", and "Identical”

The different content such as history, cookies, social networks and personal information which is gathered and turn data to **big data**, and these gathered data hen become one of the important element in extracting precious knowledge even in real time, and this can be achieved its objectives by studying these trends. The best way to evaluate the quality of **Big Data** is by its productivity and performance. [25].

Important Issues Of Big Data

As mentioned in Literature reviews, the traditional data mining algorithms are not design for parallel computing; consequently, they are not particularly useful for the big data mining. Many recent studies attempted to change the traditional data mining algorithms to make them

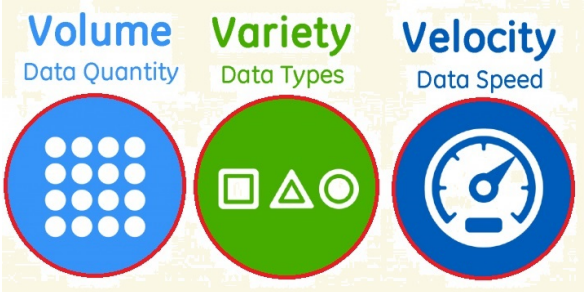


Figure 1.4 Big Data Properties

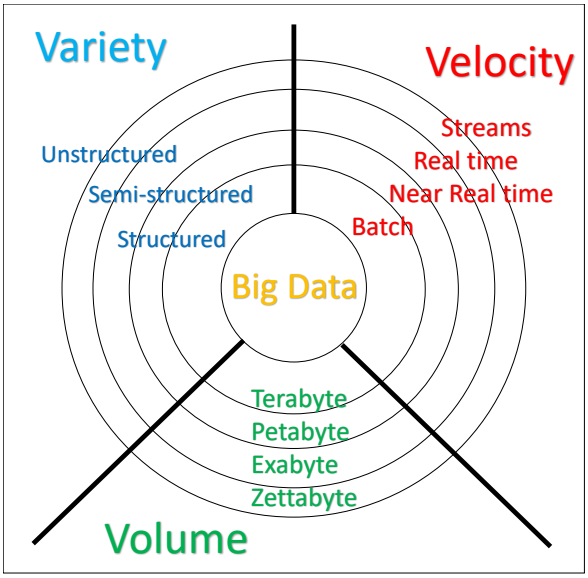


Figure 1.5 Big Data and Three Vs

applicable to big data framework such as Hadoop-based platforms. Unfortunately, most of the studies attempted failed to make the data mining and soft computing algorithms work on **Hadoop** because different algorithms are required for specific domain have the property of big data. Become more difficult to use similar framework for criminal data as it require the background in **Hadoop** and its data mining algorithms to modify or to develop new design. Another issue is that the most data mining algorithms are designed for centralized computing, and they can only work to the entire data at the same time. Therefore, it is very difficult to make them work on a parallel computing system [26].

In addition to this , also Big Data requires a revolutionary step forward from traditional data analysis, characterized in [27] ,[28],[29],[30] by its three main components: variety, velocity and volume as shown in fig. 1.5, and we will refer to it in our research as the Three Vs.

1.1.3 Dimension Reduction

Like clustering methods, dimensionality reduction algorithm such as Principal Component Analysis (PCA) explore the inherent structure in the data, but in this case in an unsupervised manner or in order to summarise or describe data using less information. This can be useful to visualize dimensional data or to simplify data which can then be used in a supervised learning method. Many of these methods can be used in classification and regression.

There are several dimension reduction procedures, but why do we need dimension reduction? considering a dataset represented as an $n \times d$ real value matrix **D**, which encodes information about **n** observations of d variables. Therefore, an important data-preprocessing procedure is to conduct dimension reduction method which finds a compressed representation of **D** that is of lower dimensions but preserves as much information in **D** as possible.

Principal component analysis (PCA) is well-known dimension reduction method. It aims to project the data to a low-dimensional orthogonal subspace that captures as much of the data variance as possible. The achievement of PCA for dimension reduction and to find the initial centroid for k-means was far better than the traditional K-means. PCA achieves the optimal result among all the linear projection methods in minimizing the squared error introduced by the projection. all the same, PCA will conduct the eigenspace decomposition on the sample covariance matrix is computational challenging when both **n** and **d** are large[31].

Principle Component Analysis (PCA)

PCA takes a dataset consisting of a set of tuples, representing points, in a high-dimensional space and finds the directions along which the tuples line up best, and The idea is to treat the set of tuples as a matrix M and find the eigenvectors for MM^T or M^TM.

Mathematically for PCA, the transformation is defined by a set of $p-dimensional$ vectors of weights or coefficients $\mathbf{w}_{(k)} = (w_1, \ldots, w_p)_{(k)}$ that maps each row vector $\mathbf{x}_{(i)}$ of X to a new vector of principal component scores $\mathbf{t}_{(i)} = (t_1, \ldots, t_l)_{(i)}$ in such a way that the individual variables $t_1, \ldots, t_l$ of t considered over the data set successively inherit the maximum possible variance from x, with each coefficient vector w constrained to be a unit Vector. A particular disadvantage of PCA is that the principal components are usually linear combinations of all input variables. Sparse PCA overcomes this disadvantage by finding linear combinations that contain just a few input variables. PCA limitation As noted above, the results of PCA depend on the scaling of the variables. A scale-invariant form of PCA has been developed.

Data mining algorithms face the curse of dimensionality problem . It is a serious problem as it will impede the operation of most data mining algorithms as the computational cost increase. Although the data analytic framework and traditional algorithms are inefficient for big data caused by the environment, devices, systems. Several open issues caused by the big data will be addressed as the platform/framework and data mining perspectives in this section to explain what dilemmas we may confront because of big data [32].

Feature selection (FS)

This section will underline the most influential dimensionality reduction algorithms according to the division established into Feature Selection (FS) and space transformation based methods. As the dimensionality increases, the computational cost increases exponentially. To overcome this problem, it is necessary to find a way to reduce the number of features in consideration. Two techniques are often used: (1) Feature subset selection. and (2) Feature extraction [33]. FS is "the process of identifying and removing as much irrelevant and redundant information as possible" [34]. The goal is to obtain a subset of features from the original data set that appropriately describes it. This subset is commonly used to train a model, with added benefits reported in the specialized literature [35] ,[36].

FS can remove irrelevant and redundant features which may induce accidental correlations in learning algorithms, diminishing their generalization abilities. The use of FS is also known to decrease the risk of over-fitting in the algorithms used later. FS reduces the search space determined by the features, thus making the learning process faster and less memory consuming. The use FS can also help in the tasks not directly related to the data mining algorithm applied to the data. FS can be used in the data collection stage, saving cost in time, sampling, sensing and personnel used to gather the data. Models and visualizations made from the data with fewer features will be easier to understand and to interpret. Maximum Relevance (MR) feature selection is a well known approach to selects m features that have a large output relevancy [37]. The feature screening method [38] is also a MR-method. MR face no challenge to apply and can be applicable to the high-dimensional data.

Nonetheless, since MR approach can only be used for the *input-output* relevance, and can not used for the *input-input* relevance, as they tend to choose the redundant features, where often we can find that the selected features from the redundant always are very similar. For that reason this method not useful in high-dimensional data.

Feature extraction (FE)

Feature extraction techniques combine the original set of features to obtain a new set of less-redundant variables [39]. For example, by using projections to low-dimensional spaces. Polynomial Expansion expands the set of features into a polynomial space. This new space is formed by an n-degree combination of the original dimensions. VectorAssambler combines a set of features into a single vector column. Single Value Decomposition (SVD) is matrix factorization method that transform a real/complex matrix M (mxn) into a factorized matrix A. The research explored large matrices, there is not for the complete factorization but only to maintain the top-k singular values and vectors. In such way, the dimensions of the implied matrices will be reduced. They also assume that n is much smaller than m (tall-and-skinny matrices) in order to avoid a severe degradation of the algorithm's performance. Principal component analysis (PCA) tries to find a rotation such that the set of possibly correlated features transforms into a set of linearly uncorrelated features. The vectors used in this orthogonal transformation are called principal components. This method is also designed for matrices with a less number of features.

In the next chapter, we will demonstrate how we used the PCA approach to assign the data points to each clusters.

1.1.4 Clustering in High Dimensional Space

The presence of too much information as in fig. 1.6 shown can make any task such as data mining quite difficult , and finding a pattern between documents, or clustering high dimensional data has lot of attention as an open research topic, For example. subspace clustering that partition the data points drawn from a union of subspaces according to their underlying subspaces, and such approaches have found widespread applications in many fields, e.g., pattern recognition, data compression, and image processing. Our research focus on social science area and specially on the criminal data analysis, and we trust this research will be an interdisciplinary approach between computer science and criminal justice.

Typically, two main branches recognised as subspace clustering approach (Top- down and the Bottom-up) which established based on their search strategy. Figure fig. 1.7 showing the Subspace Clustering Algorithms by search technique which is presented a subspace clustering algorithms map organized by the search methods and the measure used to determine locality [2].

Figure 1.6 Data increasingly massive, high-dimensional [2]

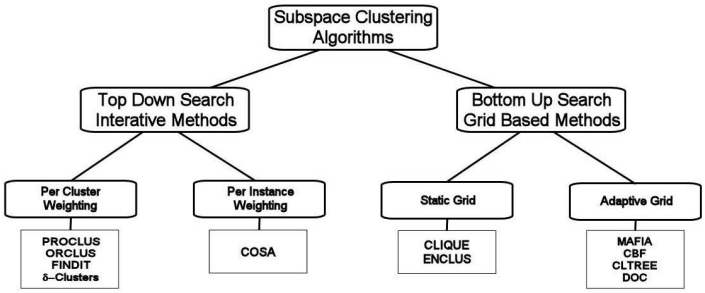


Figure 1.7 Hierarchy of Subspace Clustering Algorithms based on search methods

Top- down algorithms finding the initial clustering in the full set of dimensions and evaluate the subspaces of each cluster, then iteratively improving the results. The other approach is the **Bottom-up approaches** that to find dense regions in low dimensional spaces and then combine them to form clusters. Feature selection was one of the method been employed to improve cluster quality, and examine the dataset as a whole. These algorithms perform by removing noise and redundant dimensions.

Another reason that many clustering algorithms struggle with high dimensional data is the curse of dimensionality. when the dimensions increases, measures of the become meaningless, and dimensions expand until they equidistant from each other. The problem is become worse when the objects from one dimension are related in different ways in different subsets of dimensions.

In the graph fig. 1.8 we can see the curse of dimensionality. creating new dimension stretches the points to that dimension, pushing them further apart. which led to high dimensional data that is sparse. So the Data in one dimension is relatively tightly packed.

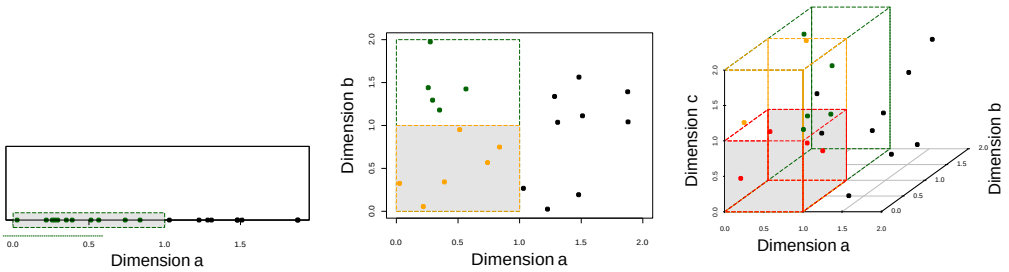


Figure 1.8 The curse of dimensionality [3]

The main problem of the high dimensionality is to find the clusters embedded in subspaces of high dimensional data, and often tackled by specifying the subspace or the subset of the dimensions which must be determined by the user [3]. The objective of subspace is to reduce the computational complexity of clustering performed on high-dimensional data.

The dimensions in big data are often lead in confusion caused by the noise data to hide clusters, and the reason is that the dimension sometimes irrelevant. Subspace clustering is solution for the traditional clustering which is look to find clusters in opposite subspaces in the same dataset. Subspace clustering algorithms find clusters in multiple overlapping subspaces by focus its search for relevant dimensions granting them to look within cluster [2].

Principal Component Analysis (PCA)

Principal component analysis (PCA) is one of the effective tool of high-dimensional data reduction. PCA is an unsupervised technique computed by the use of eigenvalue decomposition [40]. PCA is very versatile, it is the oldest and remains the most popular technique in multivariate analysis, and Its goal is to extract the valuable information from the data, to represent it as a set of new orthogonal variables called principal components and to show the match or the similarity of the variables observations as points in maps [41].

PCA includes a mathematical operation that maps a number of correlated variables into a smaller set of uncorrelated variables, called the principal components. The first principal component represents as much as possible of the variability in the data. The rest of the components describe the remaining variability. The goal of PCA [42] :

- Extract the most important information from the data table and as much variance with the fewest components

- Compress the size of the data set by keeping only this important information
- Simplify the description of the data set
- Conceptually the goal of PCA is to reduce the number of variables of interest into a smaller set of components
- Analyse variance and reduce the observed variables

Subspace Clustering (SC)

Subspace is an expansion of traditional clustering and some recognise it as an extension of the feature selection to find clusters in various spaces inside a dataset, and to reduce the computational complexity of the cluster performed on high dimensional data, by sometimes called compressed subspace clustering approach by random projection [2].

Subspace techniques has variety of applications, including text, web documents, image segmentation, medical image processing, and network data analysis, etc. Basic clustering algorithms use all dimensions of an input dataset to learn each object described [43].

In high dimensional data, yet, some of the dimensions often are irrelevant. These irrelevant dimensions sometimes called hiding clusters in noisy data, then it can be confusing once clustering algorithms applied, but Subspace methods eliminate the irrelevant, noisy and redundant dimensions by analysis of the entire dataset.

There are two branches of subspace clustering based on their search procedure [2]:

- **Top-down algorithms** find an initial clustering in the full set of dimensions and evaluate the subspaces of each cluster, iteratively improving the results.
- **Bottom-up approaches** find dense regions in low dimensional spaces and combine them to form clusters.

Figure fig. 1.9 (Left) Three subspaces, each of which is of dimension one, are sampled, where samples are represented by different markers. (Right) After being randomly projected to a two dimensional space, these samples are clustered and the structures of subspaces, denoted by lines, are revealed.

Subspace Approaches

Typically, there are two steps in subspace approaches to clustering, namely, subspace structure recovery from possibly corrupted observed data and subspace segmentation for clustering.

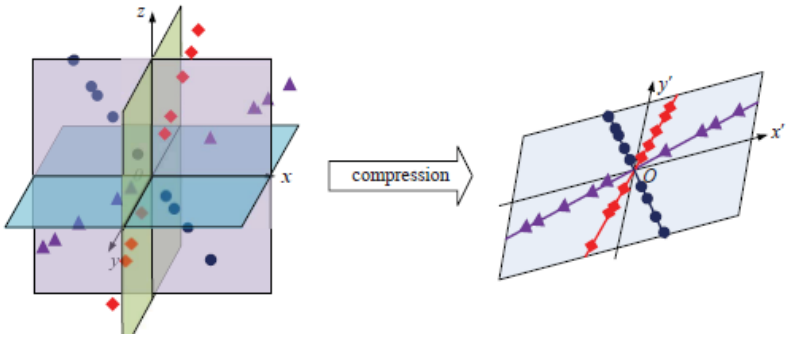


Figure 1.9 Three subspaces, each of which is of dimension one .

First, we need to extract the principal dimensions which the data have spanned, whereas next is to segment the data into clusters obeying subspace structures [44].

Clustering concept in general can be separated into two classes: Density based clustering and Graph-based clustering. High density regions in feature spaces separated by low density regions are defined as clusters [45].

The **limitation** of both types of methods is the need to build an effective similarity matrix for high-dimensional data, and in following chapter will review these limitations of these methods in details showing the proposed methods to overcome these drawbacks.

Feature Selection methods (FS)

The technique of extracting a subset of relevant features is called feature selection (FS) [46]. (FS), data sets include a large number of features. researcher mostly choose of Feature engineering methods the Feature selection (FS)to feature transformation (FT). The different between the Feature Transformation and Selection is that, the FS methods reduces the dimensionality of data by selecting only a subset of measured features to create a model, and also pick only relevant of dimensions from a dataset to make their similar groups of objects with subset of their attributes [47], but on the other hand, the FT is a group of methods that create new features, the methods of transformation are useful for dimension reduction where transformed features describe original features without loss of information features. Feature transformation methods are contrasted with the methods presented in FS, where dimension reduction is achieved by computing an optimal subset of predictive features measured in the original data.

FS can enhance the interpretability of the model, speed up the learning process, and improve the learner performance. There exist different approaches to identify the relevant features.

Model construction uses the selecting process of a subset of relevant features (variables, predictors), variable selection methods also known as Feature selection methods [48]. Feature selection plays vital role in creating an effective predictive model [49]. The reason of using feature selection is to allow the machine learning algorithm to train , and help to reduces the complexity of a model and makes it simple to interpret. In addition to this, feature selection also used to improve the accuracy of the model if the right subset is chosen, and to reduces over-fitting problem. There are several methodologies used in Feature selection to subset feature space to help models our perform efficiently: Filter Methods, Wrapper Methods, and Embedded Methods [48].

The difference between them is that, in the Filter Methods, the features are selected based of their scores of the correlation with the outcome variable, but still filter methods do not remove multicollinearity in which two or more predictor variables in a multiple regression model are highly correlated, so the multicollinearity needed to be dealt with before training models for any data. In wrapper methods, based on the inference that been drawn from previous model the subset of features used to train a model using them, as well as we could add or remove features from our subset. The difficulty is that these methods are usually computationally very expensive. The figures below illustrate the main differences between the filter and wrapper methods for feature selection [50].

These algorithms find a subset of dimensions on which to perform clustering by removing irrelevant and redundant dimensions. Unlike feature selection methods which examine the dataset as a whole, subspace clustering algorithms localize their search and are able to uncover clusters that exist in multiple, possibly overlapping subspaces [2].

Feature selection algorithms have difficulty when clusters are found in different subspaces. Feature selection clears irrelevant and redundant dimensions by investigating the whole dataset while Subspace clustering algorithms limit the search for related dimensions to discover clusters in overlapping subspaces.

- Filter methods measure the relevance of features by their correlation with dependent variable where wrapper methods measure the usefulness of a subset of feature by actually training a model on it.

- Filter methods are much faster compared to wrapper methods as they do not involve training the models. On the other hand, wrapper methods are computationally very expensive as well.

- Filter methods use statistical methods for evaluation of a subset of features while wrapper methods use cross validation.
- Filter methods might fail to find the best subset of features in many occasions but wrapper methods can always provide the best subset of features. Using the subset of features from the wrapper methods make the model more prone to overfitting as compared to using subset of features from the filter methods.

Subspace clustering need to adapt a search method that must limit the scope to consider different subspaces for each different cluster, hence, the next chapter will review the previous work and present the techniques and limitation used subspaces clustering.

Motivations

The current state-of-the-art Machine Learning techniques are difficult to understand due to the model complexity, for example a when building modules blocks for creating experiments, and each module will be accomplished by a specific machine learning algorithm, function, or code applied on data such as developing a predictive model or finding a cluster/pattern in big dataset . Finding patterns can be time-consuming, especially for large sets that changes overtime which requires additional knowledge to be discovered such as knowing how the data has changed over time.

Intelligence find it very difficult to solve crimes especially once work on multiple and massive date, indeed there already involvement from both researchers and agencies in discovering effective method that can help to predict and prevent future potential crime. So far most of techniques have been used in clustering have not reach the expectation in such domain. Even with many data mining techniques have been developed recently there is still hard to learn and find important knowledge or discovering patterns from high dimensional data, for example patterns detection never was an easy tasks for a computer data analyst or detective to identify these patterns by simple querying. Patterns cannot be discovered using the traditional visual analytic, query languages, or even some other basic clustering algorithm. The most of cluster algorithms resultant are not up to the level of the expectation when the dimensions of the dataset are high and usually kind of dataset have noisy and flawless.

This will motivate the implantation of a number of techniques of a high-dimensional data set for finding a low-dimensional space. As well as the use of AI and ML techniques in crime data analyse will make it easy to find and develop efficient clustering algorithm that will

speed up the process of solving crime for the law enforcement officers, and to bring close both researchers and agencies in discovering effective method that can help to predict and prevent future potential crime.

This study will help to meet the high level of the expectation from the cluster algorithms resultant the dimensions of the dataset are high and usually kind of dataset have noisy and flawless. The implantation of a number of techniques of a high-dimensional data set for finding a low-dimensional space.

Therefore, This research will lead to formulate crime pattern detection as machine learning task and to to support police investigator in solving crimes from complex data. as well as this research will identify the problems and obstacles that have challenged the researchers in recent years in high dimensional data reduction and specifically in clustering criminal data, and then to propose a applied method to assist many of the disadvantages of existed techniques. Let us have a glimpse on the scope of research in the area to show how the research presented in this thesis can be fitted into this diversity.

1.2 Research Problem and Approach

1.2.1 Research Questions

This research will identify the key problems of clustering high dimensional data by reviewing, studies, and discuss the related works of subspace algorithms such as Low rank representation and SC theory, and then to compare the results with previous methods that been examined on criminal data, then to proposed method for high dimensional data that will be referred to as Criminal Data Analysis Based on Low Rank Sparse Representation (LRSR) to subspace clustering algorithm.

1.2.2 Objectives

The previous approaches that has been implemented has guided us to look for the right method which has helped to come over the drawback of the traditional clustering techniques by the use of the LRR and SC of subspace clustering. The following questions are the research questions, aims and objectives, as it organised and ordered at the time where each approached has been studied, and evaluated against each problems that been review.

- Does the process of the date requires analysts, Such data may be structured or unstructured, quantitative and qualitative, of multiple formats?
- Analysing the target application and data , such as How big the data?, What is the data, and what can be knowing before clustering? and Summarising of the original set with with visual content?
- Understand the recent constraints under which the data was created ?
- Investigated the effectiveness of the proposed measure on several other real-world data sets.
- Identify the best fit and useful Clustering algorithm to start with for most of data type to identify a set of categories to describe, partitioning, and prepare the data for other AI techniques in Criminal data ?
- Identify the patterns in the relationships between data and cluster, and discover the hidden structure in unlabelled data of unsupervised learning?
- K-means implementations on such data?
- Identify the the cluster and the meaning of each group?
- What are limitations and challenges to implement k-means on such data?
- Determining the optimal number of initial point .
- Modified K-means implementations on such data?
- What data are considered as high dimensional space?
- What are limitations and challenges to implement Modified k-means on such data?
- What is process of dimensionality redundancy, and how to make better data redundancy in high dimensional space?
- How PCA preform when implemented on criminal data?
- What are the draw backs of the use of PCA segmented by k mean as criminal data redundancy ?
- What is best methods to be used as effective technique to obtain a set that could identify as the patterns in High Dimensional space?
- Why, and How to Identify the subspaces that contain clusters?

- Converting file or documents to data matrices and the time-consuming involve of convert this data frame into a matrix.
- what different between implementing the tradition PCA and the advanced data redundancy?
- What effective subspace clustering methods for high dimensional space especially for criminal datasets as rapidly changes and new information often can be added.
- How to isolate the overlapping clusters that appear in a another space?
- How to update and modify the best existing choosing approach ?
- Dose the result and techniques have fulfilled The main objective of subspace clustering helps to find high quality clusters in good time?
- The challenges and issues on how effectively to deal with the large amount of discovered patterns, and searching for useful and interesting patterns, especially when the Data come from multiple sources?
- What are the collision between New document is added, the index, and the the competing tasks are? and why is it consider as the main challenges.

1.2.3 Methodologies

The methodologies for our research based on the analytical of methods that implemented to identify the key problems and obstacles that have challenged the researchers in clustering high dimensional data. The existed subspace algorithms techniques such as LRR and SC will be reviewed, tested, and examined to be used to recover and segment embedding subspaces.

The finding and results of these methods will be discussed and compare with what already have been examined on previous approaches such as k-means and PCA segmented based on K-means. The previous approaches has helped us to chose the right subspace clustering methods to overcome their drawbacks and limitations. Both UCI Machine Learning Repository, and crime domain dataset used to find and compare the best subspace clustering algorithm that fit for high dimensional space data.

To visualize high-dimensional data in application, we used Statistics and Machine Learning Toolbox such as R, and Matlab in our experiment. Experiments are conducted to analyse the performance this new algorithms to overcome the limitation of existing subspace algorithms techniques.

1.3 Thesis Contributions

The main original contributions presented throughout this thesis theoretically and practically. The complete suggested methods have been studied and other prominent approaches in literature will be tested. besides, the methods have been studied are recently and effective in any data set, also these techniques will be helpful within the scientific community, through the publication the results of the research will be tested and applied the on our main criminal dataset as well as on the other public domain data for evaluation and comparison. There are goal-related issues have been addressed such as Missing data, Irrelevant/redundant data , Imbalanced data, and Data mining process instantiations .Theses issues it will be mentioned later for each specific data .

The experiments are conducted to analyse the effects of the proposed algorithms to come over the disadvantages of existed subspace algorithms techniques. Theses methods will be apply on both UCI Repository, and VAST database used to find and compare the best subspace clustering algorithm that fit for high dimensional space data.

1.4 Thesis Outline

This thesis is structured as follows:

- **Chapter 1** provides an introduction of AI, documents clustering, criminal data analysis, clustering in high dimensional data including PCA and subspace clustering. The second part of this chapter provides the motivation, Research question and objectives.
- **Chapter 2** provides an extensive literature survey by reviewing numerous data mining methods attempts to cluster high dimensional data, follow by the review of the related and recent work of advanced subspace clustering algorithms.
- **Chapter 3** provides an experiment of PCA segmented of k-means that has been applied to the domain of the application of VAST, and use to find the best approach for future subspace clustering in high dimensional space for criminal data analysis in future.
- **Chapter 4** provides the Low Rank subspace Sparse Representation (LRSR) model for subspace clustering, then propose an efficient scheme to solve the proposed LRSR model . The last section of this chapter is to provides introduction of two theorems regarding the convergence of analysis of the augmented Lagrangian multiplier algo-

rithms for sub-problems LRSR-1 and LRSR-2. As well as Sub-problem LRSR-1 which can be applied to future model.

- **Chapter 5** Provided our experiment we aiming to use the *LRSR* capability of clustering overlapped subspaces as well as the robustness against large noise. The method should prove the convergence of a subspace clustering framework called *LRSR*, which obtains the optimal solution based on both low rank representation (LRR) and sparse representation (SR).
- **Chapter 6** Conclusion and Future Work which concludes the thesis by giving a summary of the work and highlighting the main contributions of this research. It also underlines the limitations of the method and provides a number of suggestions for the future work. Followed by the Bibliography.

Chapter 2

Literature Review

2.1 Data Mining

“ Data becomes useful knowledge of something that matters when it builds a bridge between a question and an answer.

This connection is the signal.”

(Stephen Few) .

In statistics, pattern recognition, and artificial intelligence (machine learning), algorithms are based on the assumption that data can be loaded into the machine main memory. A perspective from databases, a field fundamental to (KDD) Knowledge Discovery, Data is provided by Imielinski and Mannila, who identify challenges posed by KDD for database technology and postulate a new direction and view for both [10].

There are a different group of algorithms and procedures that target selective domain of big data and learners. For instance, to consider Supervised learning, it could be categorized into regression and classification. When the class attribute is discrete, it is called classification; and in the case of continuous class attribute, it is regression. Decision tree learning, naive Bayes classifier, k nearest neighbor (kNN) classifier, and classification with network information are classification methods. Logistic and linear regression are regression methods. Unsupervised learning is the unsupervised partition of instances into groups of identical objects [51]. Clustering is generally categorized into three sub-domains. They are supervised, unsupervised, and semi-supervised:

Unsupervised clustering: It basically focuses on maximizing the intra-cluster similarity and minimizing the inter-cluster similarity when a similarity/dissimilarity measure is provided. It uses a specific objective function (e.g., a function that minimizes the intra-class distances to find tight clusters). K-means and hierarchical clustering are the most widely used unsupervised clustering techniques in segmentation.

Semi-supervised clustering: Along with the similarity measure, semi-supervised clustering tends to utilize other guiding domain information to enhance the clustering results. This domain information can be pair-wise constraints between the observations or target variables for some of the observations. Based on these three vital learning paradigms, a lot of theory mechanisms and application services have been proposed in order to deal with data tasks.

2.1.1 K-means Clustering Algorithm

K-means is an iterative algorithm and performs two mechanisms: first, the cluster assignment step, and second, the group centroid step, which is sometimes called the move step. The K-means algorithm must have at least the following steps or procedures:

- The first step is a cluster assignment step, which calls cluster centroids by randomly initializing the cluster's centroid vectors into two or more points. This step must always start with k centres
- The second step is a move centroid step, which is sometimes named the update step, in which the cluster centroid points that initialised in the assignment step are moved to the average of the points in the same group. In this step, we must cluster each point with the centre nearest to it.
- To compute the distance from the data vector to the cluster, the next equation was used: Where d is the dimension.

$$\mathrm{d}(\mathrm{Z_p}\,\mathrm{M_j}) = \sqrt{\sum_{k=1}^{2} (Z\ p,k - Mj,k)} \tag{2.1}$$

Where z_p is the pth data point, and M_J is centroid of i^th cluster. Each data vector calculates the distance between the data vector and each cluster centroid, which will use the minimum data vector to assign that cluster and calculate the distance using the above equation

- The centroid of the cluster is then recalculated using the following equation

$$M_j = 1/n_j(\sum Z_p)\nabla Z_p \in C_j \tag{2.2}$$

  Where n_j is the number of data point in cluster j [52].

- Repeat the previous step until the centres converges, which means that we re-run these steps until we have found no more changes.

There are two undefined aspects of the algorithm. One is the set of starting centres and the other is the stopping condition. Two inputs must be taken into consideration in order to apply the K-means algorithm. The first input is parameter K, which indicates the number of clusters we are looking for, and the second input is the K-means, which takes as the input the unlabelled training set of just the Xs, where

$$x^i \in \mathbb{R}^n \tag{2.3}$$

K-means drawback and limitation Even with the simplicity of K-means implementation, the method has many drawbacks and limitations, and not very flexible. One of the main reasons why many researchers have presented improvements to K-means algorithms where the Initial choice of the cluster number needs to be specified and known by the user [52] which called the initial centre point and the fact that there is no correct way of doing it. The other limitation of K-means is that the parameter K does not provide for external constraints. According to Singh and Bhatia in their paper [52], the Initial choice of the cluster number needs to be specified and known by the user. In the next section, we will mention in detail the K-means limitations that solved by the modified K-means.

2.1.2 Modified K-means Clustering Algorithm

Modified K-means that illustrated on paper [4] by Singh Raghuwanshi, PremNarayan Arya that compared both algorithm, they have claimed it works well with large datasets but not with high dimensional space. Modified K-means algorithm avoids getting into locally optimal solution in some degree, and reduces the adoption of cluster-error criterion [4]. Results of theier study shows with Graph, the comparison the following between K-means and Modified approach K-means on the basis of large number of records and execution time using this algorithm. Modified K-means approach better performance, comparison the figure fig. 2.1

showing the differences in the execution time in milliseconds between the Modified approach and the standard K-means algorithm.

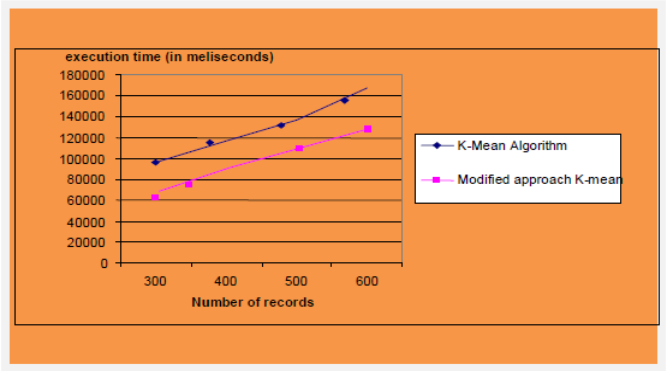


Figure 2.1 K-means and Modified approach Comparison [4]

In paper [52], a novel research on the initial point of the centroid named as the modified K-means algorithm has completed. The algorithm was constructed based on the density method of the different regions when the centroid of the cluster located in the first iteration with the maximum density of the data points.

Algorithm was applied to pick the initial centroid then enhanced the initial centroid selection by the use of the most populated space as centroid of the cluster, after dividing the space into many parts, then counted the frequency of the data points in each segment. So, the centroid calculated using the mean of each segment and locating the cluster's initial centroid. The distance calculated between each cluster's centroid was used to assign the data point to the appropriate cluster's centroid and then, for each centroid, the minimum distance calculated from the remaining centroid, and this was made into a half.

The process of the algorithm will be repeated until the data point was assigned to any of the remaining clusters. The data point has been assigned to correct the cluster's centroid; the first step involved half of the minimum distance from i^{th} cluster's centroid to the remaining cluster's centroid. The second step involved considering any data point to compute its distance from i^{th} centroid and then comparing it with $d_{C(i)}$. Then, in the third step, if it was equivalent or a smaller amount than $d_{C(i)}$, then the data point was assigned to the i^{th} cluster. Finally, the second step repeated until the end of the condition was achieved.

Modified K-means algorithm Drawback and Limitations Thus, the modified K-means approach is used to optimise the current K-means algorithm, and K-means limitation which is running the algorithm for different numbers of k values but still fail with big data and

high dimensional space. Therefore a better technique his needed for massive data and high dimensional space such as Crime dataset for better result.

Applying any clustering technique, the data must take into account the data structure and its input factors. The different between the K-means and hierarchical algorithms are that K-means clustering segments the dataset into k distinct clusters based on the distance to the centroid of a cluster, whereas hierarchical clustering uses the method of creating a cluster tree to construct a multilevel hierarchy of clusters. Consequently, either the similarity or the dissimilarity should be meaningful. Yet K-means method has many drawbacks and limitations, as well as is the initial centre point and the fact that there is no correct way of doing it, and that why many work been presented to improve K-means algorithms. These are the reasons why many researchers have presented improvements to K-means algorithms.

2.2 Dimensionality Reduction Algorithms

The most popular dimensionality reduction algorithms are Principal Component Analysis (PCA) Principal Component Regression (PCR), Partial Least Squares Regression (PLSR), sammon mapping, Multidimensional Scaling (MDS), Linear Discriminant Analysis (LDA), and Locally Linear Embedding (LLE) . Also both feature transformation and feature selection techniques included as clustering techniques for high dimensional data that represent dimension reduction in space with high dimension. As well as the subspace recognised as an extension of the feature selection to find clusters in various spaces inside a dataset, and to reduce the computational complexity of the cluster performed on high dimensional data.

Reducing the dimensionality of a vectors is generally a basic task in pattern recognition step to accomplish practical feasibility. Typically this is finished by using domain knowledge. Dimensionality decrease is additionally important in exploratory data examination, where the reason regularly is to map onto a low-dimensional space for human eyes to have an better insight and to increase some knowledge to the data [53]. The next chapter will review these techniques and their limitation.

2.2.1 Manifold Learning

Manifold learning is an approach for non-linear dimensionality reduction. Isomap [54] Similar to PCA, and is the first manifold learning method which uses geodesic distance to measure the similarity between data in high dimensional space [55]. Locally Linear Embedding (LLE) is another powerful algorithm in manifold learning. LLE supposes that

each point can be represented by a linear combination of its several neighbour points. It computes the the local combination weight matrix W in the original space firstly. Then it minimizes the reconstructed error in the new space, in which each point is reconstructed by W. [56]. However, LLE adopts k-nearest neighbors method to generate the manifold. Choosing a very small neighborhood is not a satisfactory solution, which may fragment the manifold into a large number of disconnected regions [57].

Choosing a too large neighborhood may cause “short-circuit”. Besides, target dimension also has an effect on the performance of LLE. A too small dimension can not preserve the structure and relationships in the high-dimensional space after data are mapped into a low-dimensional space [58]. LLE has strengths has the similar strengths to IsoMap which has the features of Graph-base, eigenvector method, Polynomial time algorithm, No local optima, Non-iterative, Single heuristic parameter. On the other hand LLE has also weaknesses such as sensitive to “short-cuts”, No asymptotic guarantees, and No way to estimate intrinsic manifold dimension. So, in the next section the better cluster algorithm will be reviewed to come over theses limitation.

2.2.2 Principal Component Analysis (PCA)

In cluster there are many different models in Multivariate analysis, each with its own type of analysis, as Principal components analysis (PCA) that creates a new set of orthogonal variables that contain the same information as the original set. It rotates the axes of variation to give a new set of orthogonal axes, ordered so that they summarize decreasing proportions of the variation. PCA consider as is a statistical procedure that orthogonally transforms the original n coordinates of a data set into a new set of n coordinates called principal components. Principal components analysis which multivariate clustering analysis that creates a new set of orthogonal variables that contain the same information as the original set, and then It give a new set to summarize decreasing proportions of the variation [42].

In PCA, As a result of the transformation, the first principal component has the largest possible variance; each succeeding component has the highest possible variance under the constraint that it is orthogonal to (i.e., uncorrelated with) the preceding components. Keeping only the first $m < n$ components reduces the data dimensionality while retaining most of the data information, i.e. the variation in the data. Data column ranges need to be normalized before applying PCA, and the new coordinates (PCs) are not real system-produced variables any more [59].

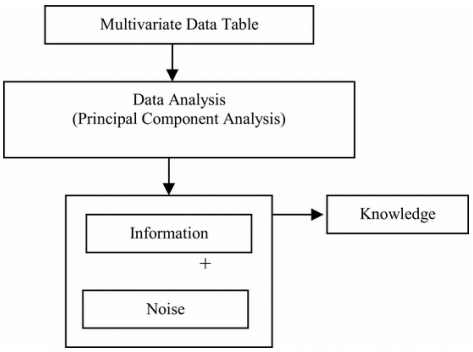


Figure 2.2 PCA concept

There are many important applications in which the data under study can naturally be modelled as a low-rank plus a sparse contribution. All the statistical applications, in which robust principal components are sought. The first principal component represents as much of the variability in the data as possible. The succeeding components describe the remaining variability. The steps involved in PCA are:

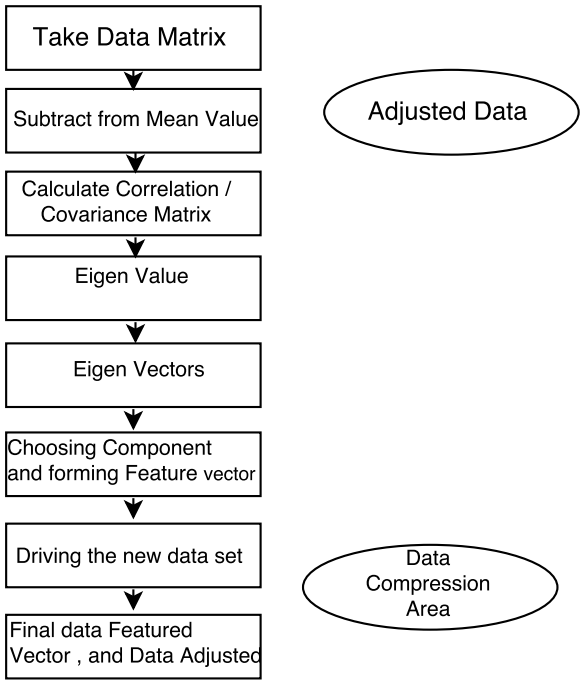


Figure 2.3 Steps involve in PCA [42]

PCA Implementation in General The eigenvectors and eigenvalues of a covariance (or correlation) matrix represent the "core" of a PCA: The eigenvectors (principal components) determine the directions of the new feature space, and the eigenvalues determine their

magnitude. In other words, the eigenvalues explain the variance of the data along the new feature axes. but Later, will explain the computational of the eigenvectors (the principal components) of a dataset that collected in a projection matrix. Each of those eigenvectors is associated with an eigenvalue which can be interpreted as the "length" or "magnitude" of the corresponding eigenvector. If some eigenvalues have a significantly larger magnitude than others that the reduction of the dataset via PCA onto a smaller dimensional subspace by dropping the "less informative" eigenpairs is reasonable [42].

1. Eigendecomposition - Computing Eigenvectors and Eigenvalues: the eigenvalues explain the variance of the data along the new feature axes, in other word , the eigenvectors and eigenvalues of a correlation matrix represent the "core" of a PCA, and the PCA which is the eigenvectors determine the directions of the new feature space, and the eigenvalues determine their magnitude.

   Covariance Matrix: The classic approach to PCA is to perform the eigendecomposition on the covariance matrix , which is a $d \times d$ matrix where each element represents the covariance between two features. The covariance between two features is calculated in following equation eq. (2.4):

$$\sigma_{jk} = \frac{1}{n-1} \sum_{i=1}^{N} (x_{ij} - \overrightarrow{x_j})(x_{ik} - \overrightarrow{x_k}) \tag{2.4}$$

   The calculation of the covariance matrix summarize via the following matrix equation eq. (2.5) :

$$\sum = \frac{1}{n-1}((\mathbf{X} - \bar{x})^T (\mathbf{X} - \bar{x})) \tag{2.5}$$

   The mean vector is a $d \times d$-dimensional vector where each value in this vector represents the sample mean of a feature column in the dataset.

   Correlation Matrix: the eigendecomposition of the covariance matrix (if the input data was standardized) yields the same results as a eigendecomposition on the correlation matrix, since the correlation matrix can be understood as the normalized covariance matrix.

   Singular Vector Decomposition: While the eigendecomposition of the covariance or correlation matrix may be more intuitive, most PCA implementations perform a Singular Vector Decomposition (SVD) to improve the computational efficiency. So, they have perform an SVD to confirm that the result are indeed the same.

2. Selecting Principal Components:

   Sorting Eigenpairs: The PCA is to reduce the dimensionality of the original feature space by projecting it onto a smaller subspace, where the eigenvectors will form the axes. However, the eigenvectors only define the directions of the new axis, since they have all the same unit length 1 . In order to decide which eigenvector(s) can dropped without losing too much information for the construction of lower-dimensional subspace, we need to inspect the corresponding eigenvalues: The eigenvectors with the lowest eigenvalues bear the least information about the distribution of the data; those are the ones can be dropped. In order to do so, the common approach is to rank the eigenvalues from highest to lowest in order choose the top **k** eigenvectors.

   Explained Variance: After they have sorted the eigenpairs, the next question is "how many principal components are we going to choose for our new feature subspace?" A useful measure is the so-called "explained variance," which can be calculated from the eigenvalues. The explained variance tells us how much information (variance) can be attributed to each of the principal components.

3. Projection Matrix: the really is that, the construction of the projection matrix that will be used to transform the Iris data onto the new feature subspace. Although, the name "projection matrix" has a nice ring to it, it is basically just a matrix of our concatenated top k eigenvectors. They have reduced the 4-dimensional feature space to a $2-dimensional$ feature subspace, by choosing the "top 2" eigenvectors with the highest eigenvalues to construct our $d \times k-dimensional$ eigenvector matrix $\mathbf{W}$.

4. *Projection* Onto the New Feature Space: In this last step we will use the $4 \times 2-dimensional$ projection matrix $\mathbf{W}$ to transform our samples onto the new subspace via the equation $\mathbf{Y} = \mathbf{X}\,\mathbf{W}$, where $\mathbf{Y}$ is a 150×2 matrix of our transformed samples.

Applying PCA The following approaches explained the summarise of the PCA steps [60] :

- Standardize the data.
- Obtain the Eigenvectors and Eigenvalues from the covariance matrix or correlation matrix, or perform Singular Vector Decomposition.

- Sort eigenvalues in descending order and choose the k eigenvectors that correspond to the k largest eigenvalues where k is the number of dimensions of the new feature subspace $K \leq d$
- Construct the projection matrix W from the selected k eigenvectors.
- Transform the original dataset X via W to obtain a $k-$dimensional feature subspace Y.

In paper [42] the dataset used to apply PCA was taken from UCI repository web site. The data set refers to clients of a wholesale distributor. The entire work is carried out using "RStudio" to simulate the algorithm. The data set refers to 440 customers of a wholesale: 298 from the Horeca (Hotel/Restaurant/Cafe) channel and142 from the Retail channel. They are distributed into two large Portuguese city regions (Lisbon and Oporto) and a complementary region. As it learned from their paper, they have used the rules for selecting the number of principal components, and the rules are:

1. Use the 'elbow' method of the scree plot (on right).
2. Pick the number of components which explain 85
3. Here retain the first three principal components.

PCA Implemented where the data set have many variables (440) , and it was be better to apply dimensionality reduction method first to make the information easier to visualize and analyse later, but as in the paper claimed there was strong correlation found in the Grocery and Detergent $-$ 0.92, Milk and Detergent $-$ 0.66 and Milk and Grocery$-$ 0.73 . There is somewhat high correlation between Channel and Detergents paper / Grocery. In this paper [42] PCA used with K–means and Fuzzy C–means algorithm. The work concludes that the computational time of PCA + Fuzzy C–means is less than PCA + K–means algorithm for the chosen application. Also, the performance of FCM algorithm is comparatively much better than the K-Means algorithm. In high dimensional space such algorithms will not work as best as subspace clustering as it will fail to find the cluster that contain in other dimension.

PCA comparing to Linear Discriminant Analysis (LDA) As we mentioned that one of the most popular dimensionality reduction algorithms also Linear Discriminant Analysis (LDA), and it is as same as the Principal Component Analysis (PCA) in its popularity. Both LDA and PCA are linear transformation methods. PCA yields the directions (principal components) that maximize the variance of the data, whereas LDA also aims to find the directions that maximize the separation (or discrimination) between different classes, which

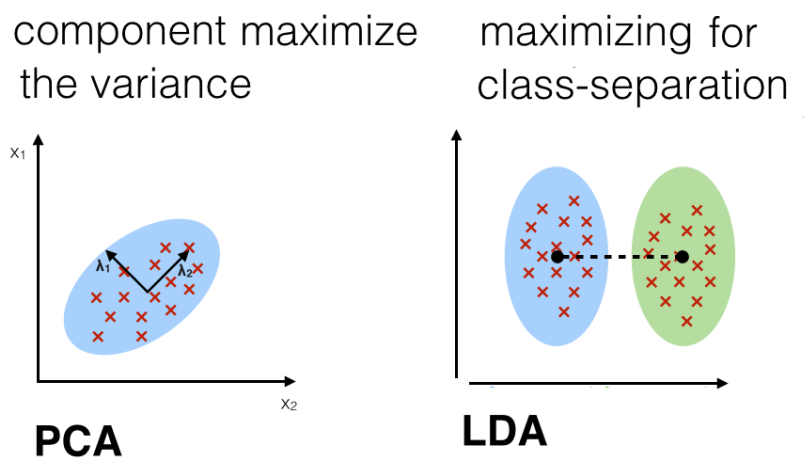


Figure 2.4 PCA comparing to LDA

can be useful in pattern classification problem (PCA "ignores" class labels). In other words, PCA projects the entire dataset onto a different feature (sub)space, and LDA tries to determine a suitable feature (sub)space in order to distinguish between patterns that belong to different classes [61].

LDA approach is very similar to a PCA, but in addition to finding the component axes that maximize the variance of data LDA interested in the axes that maximize the separation between multiple classes as in fig. 2.4 .

2.3 Subspace Clustering Approaches

Clustering concept in general can be separated into two classes: Density based clustering and graph-based clustering. Graph based clustering in fig. 2.5 showing the network of 107 journals show clusters obtained by a factor analysis on the journal-to-journal citation export matrix, and the edges are only shown between journals that have a citation pattern with a cosine greater than 0.2, and the thickness of an edge reflects the citation proximity/cosine of the connected journals. [5]

While the density based clustering class in fig. 2.6 showing the Data Driven Community Identification and Analysis , and the Data Driven Community Identification and Analysis the contrived geographical boundaries as delineated by zip codes have little relationship with the communities that grow [6].

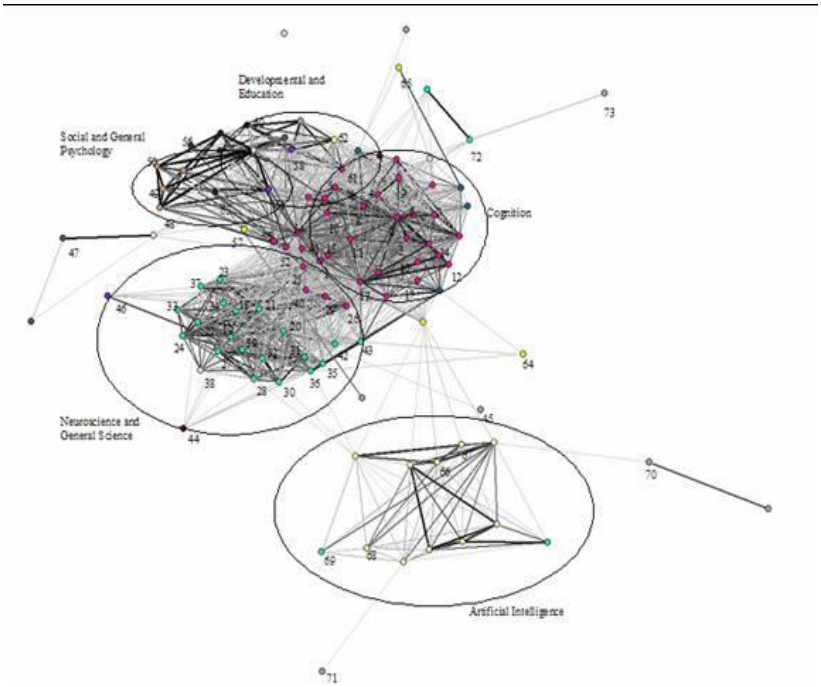


Figure 2.5 Graph based clustering [5]

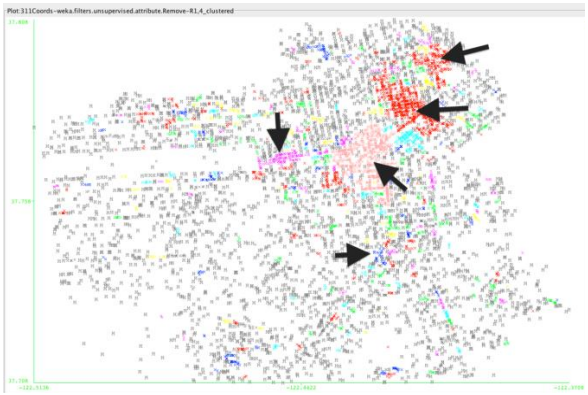


Figure 2.6 Density based clustering [6]

Density based algorithms [62], [63], [64], follow this pattern to discover density regions of clusters in feature spaces. lately, Rodriguez et al. [65] propose a strategy for Clustering by fast search and find of density peaks, and this approach based on the rule that cluster centres are characterized by a density higher than their neighbours and by large distance from points with higher densities Clusters irrespective of their shapes and outliers can be spotted consequently. Generalized Principal Component Analysis (GPCA) [66] is an algebraic method by fitting the data with polynomials. The drawback is that it is hard to estimate polynomials for GPCA when data contains large noises. Statistical approaches usually use independent samples

drawn from a mixture of probabilistic distributions to model data generation processes. It can be tackled by Expectation-Maximization (EM) algorithm with alternating between data clustering and subspaces estimation or estimating the mixture structure by iteratively finding a min-max estimation [67]. The Bayesian Ying Yang harmony learning technique [68], [69] is a unified statistical framework of modelling unsupervised learning and recent investigations [70] show that this theory can be successfully applied for determining the dimension for principal subspaces and selecting the cluster number. However, the optimization of statistical methods is difficult and the usage of the EM also leads to a local minimum.

Graph based clustering methods consist of two steps: subspace segmentation for constructing the affinity/similarity matrix and traditional spectral clustering using the affinity matrix. There are two categories: local spectral clustering and global spectral clustering methods. The typical representatives of local graph clustering based approaches are Local Subspace Affinity (LSA) [71] and Locally Linear Manifold Clustering (LLMC) [72]. For example, Zelnik et al. [73] use local information to build a similarity matrix between pairs of points. These methods are not good at handling corrupted data and sensitive to outliers. In addition, the number of neighbourhood has an impact on the performance of these methods.

2.3.1 Spectral Clustering Algorithms (SC)

Many various methods that used to process big data practitioners. Spectral Clustering Algorithms is Subspace Clustering methods, is well known to relate to partitioning of a big data, and it come under the Graph-based clustering class. SC Algorithms identified the data points as nodes in a weighted graph, and is to partition the nodes into several sets with the minimum sum of edge weights between each set. SC Algorithms is the Graph-based clustering methods that consist of two steps:

1. Subspace segmentation for constructing the affinity/similarity matrix
2. Traditional SC Algorithms using the affinity matrix. In the following sections.

Graph-based clustering such as spectral clustering [74], [75] identified data points as nodes in a weighted graph. Pair-wise dissimilarity weighs the edges between nodes. The key idea is to partition the nodes into several sets with the minimum sum of edge weights between each set. Meila and Shi [76] also propose a Markov Random Walk view of spectral clustering and proposes the Modified Normalized Cut algorithm based on the traditional K-Means algorithm, which can handle an arbitrary number of clusters.

The **limitation** of both types of methods is the need to build an effective similarity matrix for high-dimensional data.

In high-dimensional data, computing distances directly in the original space are not reliable if using the traditional methods such as k-nearest neighborhood (KNN) as the features are usually sparsely distributed. Subspace segmentation algorithms overcome the (KNN) limitation by finding clusters embedded in a low-dimensional subspace. late subspace clustering algorithms are mainly categorized into algebraic methods, statistical methods or graph-based methods.

The technique of spectral clustering is widely used to segment a range of data from graphs to images. SC Algorithms identified the data points as nodes in a weighted graph. Then Pair-wise dissimilarity weighs the edges between nodes. The key is to partition the nodes into several sets with the minimum sum of edge weights between each set. it make use of the spectrum (eigenvalues) of the similarity matrix of the data to perform dimensionality reduction before clustering in fewer dimensions. The similarity matrix is provided as an input and consists of a quantitative assessment of the relative similarity of each pair of points in the dataset. Data points often in SC as nodes of a connected graph and clusters are found by partitioning this graph, based on its spectral decomposition, into subgraphs.[77]

Spectral clustering algorithms derive from spectral graph partitioning theory . In SC, let $G = (V, E)$ represent an undirected finite graph without loops and multiple edges, $\boldsymbol{v}$ is the vertices of the graph $\boldsymbol{G}$ and the edges $\boldsymbol{E}$ connect to every pair of vertices with an associated weight . According to [78], the quality of a cluster should be determined by how similar the points within a cluster. There are some other partition , such as Minimum cut [79], Normalized cut [74], Multiway Normalized cuts and so on [80].

Notation:
Given a set of $\{x_i\}_{i=1}^{N}, x_i \in \mathbb{R}^d$ and the weight of each pair of vertices vi and measured by 8_{ij} ≤ 0,

The similarity matrix of the graph is the matrix S with ij entry Sij = 8ij . The degree of vertex is the sum of all the weights adjacent to , so it can be defined as di=ij1N 8ij , and the diagonal matrix

$$D = \begin{bmatrix} d_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & d_N \end{bmatrix} \textit{obtained}$$

2.3.2 Low-Rank Representation (LRR)

Low Rank Representation (LRR) [81], [82], [83], [84] is considered as the global approaches of seeking the intrinsic data structure. For example, Low-Rank Representation [85], [83], [86], is a general version of Robust principal component analysis (**RPCA**) [84] , which is based on minimizing the rank of the representation coefficient matrix. RPCA recovers a low-rank component and a sparse component from the observed high-dimensional data. It not only seeks the low-dimensional subspace but also handles large corruptions effectively. While RPCA take into account that data lies in a single low-rank subspace, LRR consider data from mixed subspaces by seeking the lowest-rank representation among all the candidates that can represent data well.

Theoretical analysis in [87] reveal that **LRR** is fundamentally equivalent to a work in two steps: **RPCA** to recover the subspace, followed by segmenting subspaces by the right singular vectors. However, **RPCA** lean on bringing the data points into a common low-dimensional subspace which may give rise to reduce the distances among data points of different subjects and reduce the principal angles among subspaces which may result in an affinity matrix being dense in between-classes. While LRR recovers an affinity matrix with dense within class affinities, the between-class affinities may be also dense especially when subspaces are not independent.

Low rank representation (**LRR**) [83], [86], [87] aims to find a low-rank coefficient matrix to capture the structure of data sets. When data is clean, the *sparse coding like* model is defined as follows:

$$\begin{aligned} &\min_{Z} ||Z||_* \\ &s.t. X = XZ \end{aligned} \tag{2.6}$$

It has been proven [83], [86] that the solution is given by $Z = V_X V_X^T$ where the column vectors in V_X are right singular vectors of $\mathbf{X}$. In fact, we can regard it as the least squares solution of the regression problem $X = X\ Z$. Intuitively it is very unlikely for the entries in $V_X V_X^T$ to be zero. However, **LRR** is good at seeking the global structure of subspaces and removing data outliers simultaneously. To handle corrupted high-dimensional data, **LRR** defines a nuclear

norm minimization as follows:

$$\begin{aligned} \min_{Z,E} ||Z||_* + \lambda ||E||_{2,1}, \\ s.t. X = AZ + E, \end{aligned} \tag{2.7}$$

where $||\sum||_{2,1} = \sum_{j=1}^{n} \sqrt{\sum_{j=1}^{n} ([E]_{ij})^2}$ is the $\ell_2,1$ -norm, **A** is called dictionary and the parameter λ is a trade-off the minimization between $||Z||_*$ and $||E||_2,1$.

Usually, the original data **X** have to be utilize to represent the dictionary. However, when the original data is contaminated with large noises or corruptions, they cannot represent the embedded subspaces well. Hence, it is essential to seek a clean dictionary, for which the dictionary atoms are drawn from multiple subspaces. Since $\ell_2,1$ -norm encourages the column of **E** to be zero, the underlying assumption is that the corruptions are *sample-specific*, that is to say, some data vectors are corrupted and the others are clean.

LRR has better performance than SSC in capturing the global subspace structure and protecting the within-class homogeneity. However, **LRR** assumes that data in independent subspaces is more rigorous than in disjoint subspaces.

Subspace Recovery and RPCA

High dimensional data often reside in some low-dimensional structures rather than distributed uniformly in the high dimensional ambient space. hence,recovering the low-dimensional structure can remove the disturbance of high dimensional noise and enhance the performance of clustering to a large extent.

Given that data corrupted with gross errors and outliers are ubiquitous in modern applications, how to recover clean data is essential for clustering. Robust subspace recovery is a basic problem, for which we assume that clean data set was sampled from several fixed subspaces while outliers/corruptions may be spread in the whole ambient space. One attempt is to recover the underlying fixed subspaces from the corrupted observed data. Modelling of high-dimension data with low-dimensional subspaces is the most useful paradigm in subspace recovery. Principal component analysis (PCA) is the well-known subspace recovery technique in which one minimizes the sum of squared errors of data points in the assumption that data are from a single fixed unknown subspace. However, its sensitivity to grossly corrupted observations often puts its robustness in jeopardy. That is, a single grossly corrupted entry in data could render the estimated subspace far from the true one.

A number of different approaches exist for Robust PCA, including an idealized version of Robust PCA, which aims to recover a low-rank matrix L0 from highly corrupted measurement $tsM = L0 + S0$.. Various methods like [88] [89] are proposed to reinforce the performance of subspace recovery. Among these methods, Robust principal component analysis (RPCA) [90] is ground breaking, since it provides rigorous analysis of exact low rank recovery with an unspecified

ed rank. Given a large data matrix **X**, it may be decomposed as **X** = **L** + **E**, where **L** has low rank and **E** is sparse. RPCA estimates the two components by simply minimizing a weighted combination of the nuclear norm and the ℓ_1 norm, defined as follows,

$$\begin{aligned} \min_{L,E} ||L||_* + \lambda ||E||_1, \\ s.t. X = L + E, \end{aligned} \tag{2.8}$$

where $X \in \mathbb{R}^{MXN}$, **L** denotes the matrix of clean data lying in a low dimensional subspace and **E** can be considered as the deviation of **X** from the intrinsic low dimensional subspace. For video sequences, **E** represents moving objects in the low-dimensional background. To solve the optimization problem 2.8, ADM [91] achieves much higher accuracy and better convergence performance than other algorithms [92], [93]. The main cost of these algorithms is Singular Values Decomposition **(SVD)** in each iteration. However, RPCA cannot be directly applied to clustering since RPCA assumes that data are drawn from a single subspace. Therefore, RPCA tends to bring data points into a common low dimensional subspace which may result in lessening the distances between data points of different subjects and the principal angles between subspaces in a practical application.

2.3.3 Subspace Segmentation (SS)

Subspace segmentation refers to the problem of segmenting high-dimensional data according to their underlying subspaces.

Let $\{S_e\}_\ell^n =_1$ be a set of n linear subspaces with dimensions $\{S_e\}_\ell^n =_1$. A given collection of data points $X = [x_1, x_2, \ldots, x_n]$ lies in the union of n unknown subspaces. The sparse subspace clustering proposed in [7] relies on the so called data self-expressive property in the sense that each data point in a union of subspaces can be efficiently expressed as a linear combination of all the other data points, i.e. , $x_i = X_{z\ i}, z_{i\ i} = 0$. And the sparsity penalty will

drive this expression to be only dependent of data points in the same subspace as the data point x_i . In the ideal scenario for the whole dataset, we wish to recover a coefficient matrix $\mathbf{Z}$ of $\mathbf{X}$ which has $Block - Diagonal$ structure.

Independent Subspaces A collection of subspaces $\{S_i\}_i^n = 1$ is said to be independent if $\dim(\oplus_i^n =_1 S_i) = \sum_i^n =_1 \dim(S_i)$, where $\oplus$ denotes the direct sum operator.

Disjoint Subspaces A collection of subspaces $\{S_i\}_i^n = 1$ is said to be disjoint if for every pair of subspaces we have $\dim(S_i \oplus S_j) = \dim(S_i) + \dim(S_j)$. Independent subspace condition is much stronger than disjoint condition especially when there are more than two subspaces.

Block-Diagonal Property of the Coefficient Matrix Given a data set $\mathbf{X}$ drawn from n disjoint subspaces $\{S_1, \ldots, S_n\}$, we assume data points within a subspace are indexed sequentially, $\mathbf{X} = [X_1, \ldots, X_n]$ such that $X_l \subset S_l$ (for $\iota = 1, 2, \ldots, n$). The data self-expressive property gives

$$\mathbf{X} = [X_1, \ldots, X_n] = [X_1 Z_1, \ldots, X_n Z_n] = \mathbf{XZ}$$

When the coefficient matrix $\mathbf{Z}$ has the following structure

$$\mathbf{Z} = \begin{bmatrix} Z_1 & 0 & \ldots & 0 \\ 0 & Z_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & Z_n \end{bmatrix}, \tag{2.9}$$

we say that the data set $\mathbf{X}$ satisfies the *block-diagonal* property. For example, there are several experimental results of the synthetic data drawn from disjoint subspaces and contaminated with large noises, and we will evaluate the proposed method LRSR recovers a relatively nice Block-Diagonal structure of the data.

2.3.4 Sparse Subspace Clustering (SSC)

Sparse Subspace Clustering (SSC) is the result of the development of compressive sensing in signal processing, (SSC) [7], [94], [95] apply features of sparse representation (SR) for subspace clustering. Elhamifar and Vidal [7] implemented subspace segmentation based on finding the sparsest representations for the data set.

According to the theoretical work [7], [94], [95] the subspace-sparse recovery holds when the smallest singular value measuring how well representing each subspace is smaller than the smallest principal angle among other subspaces. However, the search of sparsity separates

within cluster points in the same cluster into several small singletons particularly when some data points are corrupted.

Sparse subspace clustering aims at revealing disjoint subspaces from data by seeking the sparsest representation solution as follows:

$$\begin{aligned} &\min_{Z,E} ||Z||_1 + \lambda ||E||_1, \\ &s.t. X = XZ + E, diag(Z) = 0, \end{aligned} \tag{2.10}$$

where **E** corresponds to a matrix of sparse error entries. In the noise free cases, Elhamifar and Vidal [7] proves that when the distribution of data in each subspace and the least principal angle with each of other subspaces satisfy certain Conditions, then the following ℓ_1 minimization can be successful in subspace sparse recovery, i.e.,

$$\begin{bmatrix} z^* \\ z^*\_ \end{bmatrix} = \arg\min \left\| \begin{bmatrix} Z \\ Z\_ \end{bmatrix} \right\|_1 \textbf{s.t. } x_i = [\mathbf{X}_i \ \mathbf{X}\_i] \begin{bmatrix} Z \\ Z\_ \end{bmatrix},$$

where x_i is in the subspace spanned by $\mathbf{X}_i$ and the solution satisfies $z^* \neq 0$, $z^*\_ = 0$. After solving z_i for each data point x_i and normalizing the columns of $\mathbf{Z}$, $\mathbf{W} = (|Z| + |Z^T|)$ can be used as an affinity matrix for spectral clustering. SSC has a good performance for disjoint subspaces or subspaces with non trivial intersections if the principal angles between subspaces are not too small [96], [97] .

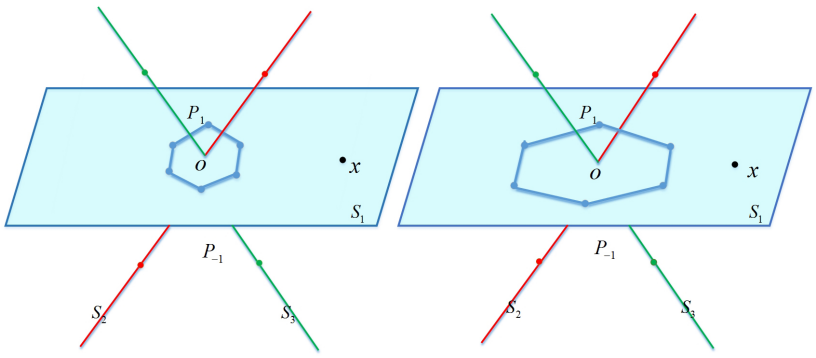


Figure 2.7 Disjoint subspaces [7]

Figure 2.7 Disjoint subspaces $\{S_1, S_2, S_3\}$: The black dotted line denotes the subspace inter section of $S_1 and S_2 S_3$. Left: the "Polytope" [1]

[1] "Polytope " subset P of Rd is called polyhedron if it is the set of solutions to a system of linear inequalities, and called convex polytope if it is a convex polyhedron and bounded. When a convex polyhedron and call

that arising from defied geometric computation in general dimensional space, thus polytope $\gamma_1 P_1$ reaches x later than that of $\gamma_1 P_1$ when appropriately increasing both $|\gamma_1|$ and $|\gamma_- 1|$, hence a sparse recovery does not hold for this case. Right: The smallest singular value of S_1 is bigger than the smallest principal angle between S_2 or S_3, then the polytope ${}_1P_1$ reaches x for a smaller $|\gamma_1|$ than that of i.e., x can be represented by $\gamma_1 P_1$. Hence the sparse recovery holds. See [7].

Figure 2.7 shows that if the smallest principal angle of $S1$ between S_2 and S_3 is larger than a certain value related to the data distribution in S_1, then the sparse representation criterion can guarantee a coefficient matrix with strictly $diagonal - block$ structure [7], . However, in actual situations, SSC always *over-segments* datasets by constructing a sparse within-class affinity matrix especially when data is greatly corrupted. Thus, the sparse representation $\mathbf{Z}$ is actually not a low-rank one. On the other hand, SSC is not good at removing outliers, thus it often has bad performance when data are greatly corrupted. In fact, low rank property is more desired for the purpose of clustering. For example, a diagonal matrix is sparse but not in low rank, then it is not meaningful for clustering.

Based on the advantages of subspace clustering and to improve the performance of these methods, we will propose subspace segmentation method Low Rank subspace Sparse Representation (LRSR) which not only recovers the low-rank subspaces but also get a relatively sparse segmentation with respect to disjoint subspaces or even overlapping subspaces.

2.4 Criminal Data and Big Data Analysis

"Advances in technology, which allow analyses of large quantities of data,

are the foundation for the relatively new field known as crime analysis."

(Deborah Osborne, [99])

Crimes are unpleasant social activity and as can damage any society if exist in several ways. In the past solving crimes has been the privilege at the criminal justice and law enforcement, and to solve , identify , or track crime become the reason for growing use of the technologies. The crime dataset often either kept private or public information, but information about the

them simply polytopes and polyhedra as shown in fig. 2.7. However, In many applications involving convex polytopes, what is most important is the combinatorial type of the polytope of how many faces are there in each dimension, and which faces are incident. [98]

offender often only viewed by authorities , which make the study of such domain more sensitive and complicated in nature [100].

The Big Data spans across three dimensions: Volume, Velocity and Variety as shown in fig. 1.4 . There characteristics of Big Data are [101]:

- **Volume**: often refer to the size of data, and this size often is very large and calculated in Terabytes or Petabytes. In addition to this, the big data can be explained in the following terms of:
  - Records per Area
  - Transactions
  - Table
- **Velocity** : used to describe the measure of how fast the data is rated at which it is being generated, for example 250 billion users to upload their images, and there are Various applications based on data rate are:
  - Batch: Batch means running the query in a scheduled and sequential way without any intervention. Execution is on a batch of input
  - Real Time: Real time data is defined as the information which is delivered immediately after its collection. There is no delay in the timeliness of information provided.
  - Interactive means executing the tasks which require frequent user interaction.
  - Streaming: The method of processing the data as it comes in is called streaming. The insight into the data is required as it arrives.
- **Variety** : It extends beyond the structured data, including unstructured data of all varieties such as text, audio, video, posts, log files etc.

Big data known for massive data sets that has large, and more varied and complex structure that led to difficulties in analysing, storage, and visualizing. As well big data need advanced ML methods for better further processes or results. The process of investigating into big data help in discovering hidden patterns and relations. For this fact, big data implementations require the use of technicalities for an insights that open new sources of research value. There are three characteristic of big data: volume, variety, and velocity as in fig. 1.5. The volume of the data is its size, and how enormous it is. Velocity refers to the rate with which data is

changing, or how often it is created. Finally, variety includes the different formats and types of data, as well as the different kinds of uses and ways of analysing the data [102].

2.4.1 Definition of The Domain Problems

There are a few objectives to comprehend the domain problem in big data analysis. first is to understand the basic of the dataset, by identifying ideas and connections that are not represented clearly in the dataset, but rather are a vital piece of any clarification of what the dataset about or how it was evolve. An increasingly extraordinary objective may be to build a total arrangement of necessities, for example, data quality and data pre-processing [103] .

However, there are critical questions to be answered ahead of making decision of which ML methods to select, as it is significant to evaluate what kind of problem we try to be solve? e.g. is it classification, clustering, or dose data have very high dimension space, also is the data labelled? It's necessary tasks that equally important as applying , improving and evaluating any advance ML algorithms such as subspace clustering algorithms.

2.4.2 Big Data Knowledge Discovery and Data mining

Data mining assist in patterns discovery of observed data. Two primary mathematical approach [104] are used in model fitting: statistical and logical. Knowledge Discovery from Data (KDD) designed to get information from complicated data sets [105]. Reference [10] outlines the KDD at the following steps:

- Application domain prior to information and defining purpose of process from customer's perspective.
- Generates subset data points for knowledge discovery.
- Removing noise, handling missing data fields, collecting required information to model and calculating time information and known changes.
- Finding useful properties to present data depending on purpose of job.
- Mapping purposes to a particular data mining methods.
- Choose data mining algorithm and method for searching data patterns.
- Researching patterns in expressional form.

Table 2.1 DATA MINING TECHNIQUES AND THEIR ROLES

Techniques	**Roles**
Classification	Pre-Defined Examples
Clustering	Identification of similar classes of objects.
Prediction	Regression Technique.
Association Rules	Find frequent item set findings among large data sets.
Neural Networks	Derive meaning from data to extract/detect patterns that are complex.
Decision Trees	set representation of decisions using Classification and Regression Tree
Nearest Neighbour method	Classify dataset records based on the K-records most similar .

- Returning any steps **1** through **7** for iterations also this step can include visualization of patterns.
- Using information directly, combining information into another system or simply enlisting and reporting.

Data mining and knowledge discovery in databases are similar to each other and to other related fields such as statistics, ML , and databases. Data Mining is one of the step process of KDD that consists of collection and preprocessing of data, data mining, interpretation, evaluation of discovered knowledge and finally post processing [106].

Understanding of the Big data

The KDD primary aim is to achieve meaningful of the dataset through the development of new methods of data mining by the KDD process is to map immense and heterogeneous data into understandable, and more in useable form [104], [107]. There are different data mining techniques to extract information from a dataset and transform it into an understandable format for further use. table 2.1 shows different Data Mining Techniques and their roles.

Important goals of data mining are predictions and descriptions. Description to discover patterns describing the data where predictions to predict unknown values of other variables of interest [108], [109] . In [110] been proposed methods that learning function to maps a data item into one of several predefined categories refer to it as classification. On the other hand Apte and Hong in [111] recommended that classification technique of data mining are used same as knowledge discovery applications. Where regression it is result of that classifier because regarded as a predictive method than classification problem as it maps data point to a prediction variable. However clustering known as descriptive method where it does identify

set of categories that describe the data [112], [113],[114]. Summarization include methods similar to calculating the mean and the standard deviations. There are some methods that as well include the abstract rules derivation , visualization method, and functional discovery of relationships amongst variables [104], [107]. Also summarization method often used for an interactive exploratory of dataset for the purpose of analysis and an automated generation.

The comparison from [8] shows the different between machine learning paradigms to overcome big data challenges is display in fig. 2.8 that illustrate the comparison of various big data parameters targeted by machine learning paradigms.

	APPROACHES	CHALLENGES																	
		VOLUME								VARIETY			VELOCITY				VERACITY		
		Processing Performance	Curse of Modularity	Class Imbalance	Curse of Dimensionality	Feature Engineering	Non-linearity	Bonferonni's Principle	Variance and Bias	Data locality	Data Heterogeneity	Dirty and noisy Data	Data availability	Real-time Processing/Streaming	Concept drift	I.i.d	Data Provenance	Data Uncertainty	Dirty and Noisy Data
LEARNING PARADIGMS	Deep Learning					✓	✓				✓								
	Online Learning	✓	✓							✓			✓	✓		✓			
	Local Learning	✓	✓	✓					✓	✓									
	Transfer Learning			✓							✓								
	Lifelong Learning	✓		✓							✓		✓	✓					
	Ensemble Learning	✓	✓												✓				

Figure 2.8 Comparison of various big data parameters targeted by machine learning paradigms [8]

Dealing with the Big Data

Data preparation is an important part of the machine learning process, as it helps to improve the performance of the operators of data analytic which is the part of KDD process to finding the hidden information in the data, such as data mining [24]. It is a monotonous process, a very significant part of data preparation is to assess the quality of any dataset. Nowadays, the data that need to be analyzed are not just large, but they are composed of various data types, and even including streaming data [26].

To make the whole process of knowledge discovery in databases (KDD) more clear, Fayyad and his colleagues summarized the KDD process by a few operations in [115], which are selection, preprocessing, transformation, data mining, and interpretation/evaluation. As shown in Fig. fig. 2.9 , with these operators at hand, we will be able to build a complete data analytic system to gather data first and then find information from the data and display the knowledge to the user .

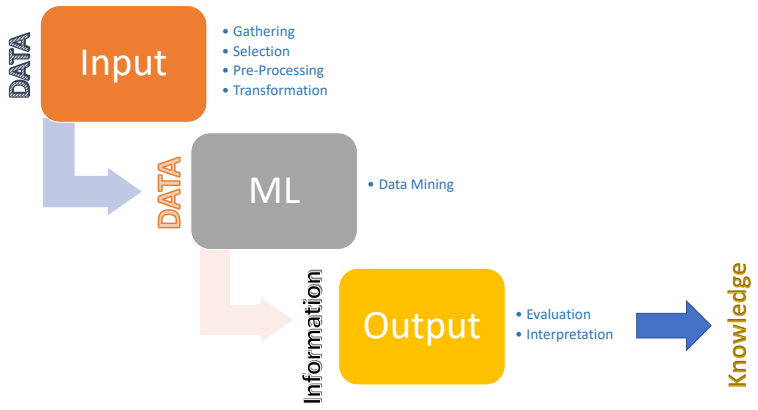


Figure 2.9 The process of knowledge discovery in databases

2.4.3 Data Quality and Preparation

Data preparation is a cardinal step in data analysis. As a large amount important/unimportant of information is available in dataset, many are interested in transforming these information into quality dataset that can be used for further purposes. This bring forth the urgent need for data analysis. as well poor quality data probably result in inconsistent quality of insights to apply data mining or for visualisation implementation. The quality data help in better transformation process and able to change the dataset into useful KDD and information [116].

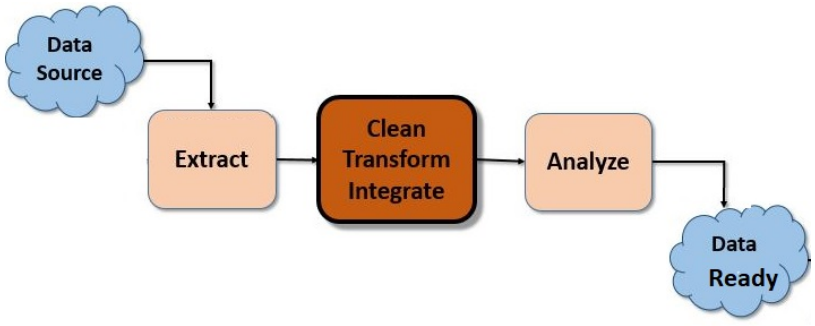


Figure 2.10 Data Quality and Preparation Time consuming

Data quality and preparation its very long and challenging tasks. If we look at fig. 2.10 we can see the task of data clean and preparation time often could be from **60** % up too **80** % from th total of the project time . Data visualization can be a part of the analyse or present stages.

Data pre-processing is necessary task for acquire quality outcome from the knowledge discovery algorithm under condition [117]. Data cleaning, data integrity, high dimension are the main concerns in preparing the data for analysis. As the dimension increases the computational cost related likewise will increases exponentially. To address this problem it is very much necessary to reduce the number of features to be considered [118]. One solution is to use feature selection and extraction algorithms. There are two approaches for dimension reduction first is the Feature subset selection and the second approach is the Feature Extraction. In next section will review each of these techniques in more details

Big Data Challenges

The main challenges of **Big Data** are data variety, volume, and Velocity. Many are struggling to deal with the increasing volumes of data. There are many proposed techniques In order to solve these issues, such as the need of reducing the amount of data being stored without losing the value of the data to help for prediction and for finding the patterns. The difficulty often related to data capture, storage, search, sharing, analytic or visualization etc. But The main goals of high-dimensional data analysis is to develop effective methods to accurately predict any future observations, as well is to discover the relationship between any features and response for scientific purposes. What are the challenges of **Analysing Big Data?** Big Data are characterized by high dimensionality and large sample size. These two features raise three unique challenges: (i) high dimensionality brings noise accumulation, spurious correlations and incidental homogeneity; (ii) high dimensionality combined with large sample size creates issues such as computational cost and algorithmic instability; (iii) the massive samples in Big Data are typically aggregated from multiple sources at different time points using different technologies. This creates issues of heterogeneity, experimental variations and statistical biases, and requires us to develop more adaptive and robust procedures [31].

2.4.4 Feature Engineering

Coming up with features is difficult, time-consuming, requires expert knowledge.

"Applied machine learning" is basically feature engineering.

(Andrew Ng, [119]) .

Many machine learning models are algebraic, thus their input must be numerical, and there are several models used to transformed non-numerical data. n machine learning projects, the focus shifts to representation than to focus on coding as programmer. A high dimensional

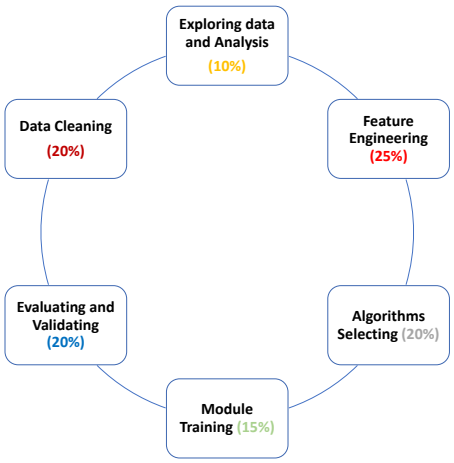


Figure 2.11 Data scientists time consume

data often its feature space is solved by feature selection and feature extraction methods to improves the performance of machine learning algorithms goal such as automated feature learning, this called feature engineering. Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work, and to create new input features from your existing ones. The feature selection and feature extraction techniques remove the irrelevant features from the text documents and reduce the dimensionality of feature space [120].

Data scientists usually spend the most time on exploring the data, cleaning the data, and Feature engineering more than in algorithm selection, training , and evaluation as in fig. 2.11.

Features: A feature is an attribute shared by all of the independent units on which analysis or prediction is to be done. Any attribute could be a feature, as long as it is useful to the model [121]. The process of feature engineering is:[122]

1. Brainstorming or Testing features;
2. Deciding what features to create;
3. Creating features;
4. Checking how the features work with the model;
5. Improving the features if needed;
6. Go back to brainstorming/creating more features until the work is done [122].

A. Feature Selection

"Feature selection is itself useful, but it mostly acts as a filter,

" muting out features that aren't useful in addition to your existing features.

(Robert Neuhaus)

Feature selection has become a essential before applying any classification methods, but Unlike feature extraction methods, feature selection techniques do not change the original representation of the data [123]. A subset from available features data are selected for the process of learning algorithm and its often in feature selection process. The **best** subset is the one with **least number** of dimensions that most **contribute** to learning accuracy [124]. Both feature subset selection and feature extraction methods have same objective which is to avoid overfitting. Feature selection in this case are considered the simplest method, in which the number of features is reduced by selecting only the most meaningful and important features according to their criterion for example as high levels of activity.

Feature selection has been used on various ranges of data, including both low and high dimensional data, although it is primarily utilized with high dimensional data to remove redundant and unwanted features. The methods of feature selection are $filter$, $wrapping$, and $embedded$. These are pre-processing techniques to analyse the benefit of features, disregard the effects of selected feature subset on performance of learning algorithm. Performing a feature selection to get knowledge discovery, interpretability and to gain some insights, and to overcome the curse of dimensionality. Feature selection methods can be used to identify and remove unneeded, irrelevant and redundant attributes from data that do not contribute to the accuracy of a predictive model or may in fact decrease the accuracy of the model [120] .

table 2.2 [9]. $filters$ extract features from the dataset without any learning applied, and suitable for very large features, only the accuracy is not secured, and evaluates the value of features without using any learning algorithm, also evaluates the features using formula based on the attribute of data [125], [126]. $Wrappers$ uses ML techniques to evaluate most useful features [127], accuracy measured of wrappers approach by the algorithm is really high [128], and to assess the value of features by employ a predetermined learning algorithm to the dataset to seek quality of the selected subsets [126], [129]. This approach is best accuracy, merely have high computational complexity and lacks generality. $Embedded$ method have best computational complexity than wrapper approach, which combine the feature selection step and the classifier construction and it consider as greedy algorithms based on (SFS) sequential feature selection which fail to find feature set that maximize the

Table 2.2 Feature subset selection Algorithms[9]

METHODS	ALGORITHM	FEATURES
Filter	FOCUS RELIEF	Suitable for large features
Wrapper	Forward selection Backward elimination	High accuracy
Feature subset selection	FAST	Improves classifier performance
Embedded	Sequential Forward Selection(SFS)	Forward Selection Interacts with the classifier Better computational complexity. Models feature dependencies.

measures independence between two variables, and in this case between features vector and the labels. This failure due to the sparsity of $high-dimensional$ data. Evaluating between two scalar variables is feasible through histograms, and this approach has found use in feature selection rather than in feature transformation [130] [131] [132]. In concept, finding a transformation to lower dimensions might be possible than selecting features [133].

B. Feature Transformation

Feature transformation is a group of methods that create new features called also predictor variables. Feature selection is a subset of feature transformation. While some machine learning packages might transform categorical data to numeric automatically based on some default embedding method, many other machine learning packages don't support such inputs. There many other methods to use to prepare the data in specific ways before fitting a machine learning model. Principal component analysis (PCA) it is well known but has nothing to do with discriminative features optimal for clustering, since it is only concerned with the covariance of all data regardless of the class. Rotation is a simple linear transformation. Several preprocessing methods such as principal component analysis (PCA) perform such linear transformations, which permit reducing the space dimensionality and exhibit better features [134]. Rotations in feature space often simplify feature selection. However, it may be very useful in reducing noise in the data. So called linear discriminant analysis (LDA) can be used to derive a discriminative transformation that is optimal for certain cases [133].

Categorical Transformation , Categorical data need to be transformed as some algorithms cannot work with categorical data directly because cannot operate on label data directly, and

they require all input variables and output variables to be numeric, and in this problem of converting categorical data to **Numerical Data** there three mutual methods [135] :

1. One hot encoding (OHE)
2. Feature hashing (FHE)
3. Encoding to ordinal variables

All of these methods above are used either with $Random-Forest-Loglolss$ or $Logistic-Regression-Loglolss$, thus we do not wanted to use the utility function to create dummy variable by encoding to ordinal variables which is converting the category into numeric values as It is inappropriate to use the encoding to ordinal variables as this method will not make any sense to code $first$ value = 1 and the $second$ = 2 ..etc. , it is just random and it will assume that these values are similar to another relying to the value number only rather than encoding each category as a one hot encoding (OHE) vector (or dummy variables) [135] .

2.4.5 Feature Extraction

In general, features can be as relevant, irrelevant, or redundant. Feature extraction aims is to choice features as a pre-processing. A subset from available selected features data are for the process of learning algorithm. The best subset is the one least number of dimensions that most contribute to learning accuracy. There are four expression of feature extraction: feature construction, feature subset generation, evaluation criterion definition such as relevance index or predictive power), and evaluation criterion estimation . The feature construction is independent technique but the rest of the aspects are related somehow to feature selection. $Filters$ and $wrappers$ distinguish by the evaluation criterion. It is usually understood that $filters$ are used criteria not involving any ML, whereas $wrappers$ use the performance of a ML trained using a given $feature_subset$ [134].The Pearson correlation coefficient is a individual feature ranking, and it is also closely related to the $T-test$ statistic, and the $Naßve\_Bayes$ ranking index.

However, the feature extraction step followed by clustering on feature vectors in reduced-dimension space, and the feature vector is just a vector that contains information describing an object's important characteristics. Feature extraction and dimension reduction can be combined in one step using; principal component analysis (PCA), linear discriminant analysis (LDA), canonical correlation analysis (CCA), or non-negative matrix factorization (NMF) techniques [136]. Feature subset selection apply by removing features that are not relevant

or are redundant. The subset of features selected should give the best performance according to some objective function [137].

2.5 Summary and Discussion

Data mining technique are used to analyse data and extract useful information from large amount of data ,and KDD is the methods development to find the relevant in any dataset. To predict or to cluster data mining techniques used as part of the most KDD software and visualization. KDD give rise in good decision-making. Machine learning is an ideal tool for extracting the hidden patterns in the data and making efficient predictions on the same. One of the primary advantages of this paradigm is the minimal dependency on the human factors that make it to deliver its best among disparate and wide variety of sources. It is powered by data running at the machine scale. It performs well when the data is to be dealt with is large in volume, high in speed and diverse in variety. Unlike conventional analysis of data, machine learning thrives with growing data. The more data is entered into a machine, the more it can learn and apply the results for advanced quality insights.

Also, there often only three groups who are highly interested in the Big Data topic; First, is the Machine Learning algorithm and Data scientist group who interest is to approach the problem to find the optimize model to get perfect performance on the data based on the income and on what they have considered, where the second group who under the AI and (IoT)Internet of Things where they focused more on the impact that cloud migration and streaming will have on big data implementations and then how they will set their goals. The third group whom focus more on businesses side more than the research view, for example books that explores the consequences and benefits of the expanding big data and what are the benefits of big data growing at an astronomical rate. the Big Data often have one of the three main characters, **big quantities** refer to Volume , **high speed** of which data been generated called Velocity, and **different types** of data refer to it as Variety.

The dimensionality reduction often does not obstruct the performance of classification algorithms. Whereas with reduced number of transformed feature exponential reduction in computation time can be observed for Big data. nonetheless, the algorithm rely on the character of data being processed. Pre-processing of Big data algorithms for preparing better representative feature sets highly needed for better predictions, as it essential for prediction of data analysis.

Some argues that to achieve least number of dimensions is best to reduce data and PCA best example but applying PCA apply a PCA before in order to reduce the dimensionality of features, we may conserve the most cumulative percentage of inertia (98-99 % . Moreover, depending of our features, maybe we must not use PCA at this stage. you should perform feature selection for relevant subset , i.e. discard the features that are little or not relevant for discrimination. PCA takes care of input space only, hence does not say anything about the relevance of your features for classification. PCA may provide a compact repesentation of your input data by finding linear combination of the features; but if the features are irrelevant, linear combinations of them will not help. Therefore, you should first perform feature selection, and, once you have discarded irrelevant features, perform PCA on them. PCA may, or may not, be useful, depending on the geometry of the data in the space of relevant features.

To apply any clustering technique, the data must take into account the data structure and its input factors. The difference between the K-means and hierarchical algorithms are that K-means clustering segments the dataset into k distinct clusters based on the distance to the centroid of a cluster, whereas hierarchical clustering uses the method of creating a cluster tree to construct a multilevel hierarchy of clusters.

Consequently, either the similarity or the dissimilarity should be meaningful. Yet K-means method has many drawbacks and limitations, as well as is the initial centre point and the fact that there is no correct way of doing it, and that why many work been presented to improve K-means algorithms. These are the reasons why many researchers have presented improvements to K-means algorithms. Therefore, Clustering will be in general a perfect choice and Subspace Clustering in particular is the best technique to exploit that can further improve our application performance. So, in our Advanced LRSR for Big Data application section we will show how our domain been consider as big data domain and how we have tackled the issues related to such domain and how we have reached our the optimal module for Big Data. The aim of this research is to present a comparative analysis of the Machine Learning algorithms to best reconcile Big Data challenges drawn on the basis of optimized performance with respect to time and accuracy obtained in analytic and in prediction.

In this section we have introduced what is the big data, big data analysis, and challenges. In next chapter , we will test and evaluate these methods against the methods on two real-world datasets.

Chapter 3

PCA Segmented K-Means Clustering

3.1 Chapter Overview

In this chapter [1] where a similar crime has happened over a period of time, it is possible to manage law enforcement resources more effectively like assisting to identify suspects. will show our experimental of the approaches and methods of PCA segmented of K-means that has been applied to a dataset of a crime domain . besides a summary and discussion will be provided, which has led us to find the best approach for future subspace clustering in high dimensional space for criminal data analysis in future.

3.2 Background of the Application

The outcome will be obtained from the record, which was a result of tracking the park visitors' information over three days. This will allow us to achieve various hidden predictive patterns by analysing all the observations from large VAST databases. One incident occurred at the park during the weekend. A crowd had gathered at the park to honour a local soccer celebrity. Local police appeared on the scene shortly after the park visitors discovered some vandalism. Security guards were questioned to eliminate the possibility of an inside job. Visitors use park applications to check-in, to go on rides, and to communicate with fellow visitors. If visitors do not have compatible phones, they are provided with loaned devices. Visitors are assigned IDs and must use the app to check into rides and other attractions. The park equipped with sensor beacons that record movements within the park. The sensors each

[1] Published at IEEE 2016 International Conference on Machine Learning and Cybernetics (ICMLC)[138]

cover a 5m x 5m grid cell. All the pathways are covered by these sensors including both the park and all the ride check-in locations. Locations are not recording while people are on rides or inside attractions, including restaurants, stores, and restrooms. App users may send text messages to anyone within their preferred group (for example, a family could have their private group). An app user may also make a friend at the park and can send and receive texts if both persons accept the friend invitations.

3.2.1 The Data Values

VAST data has two datasets, Movement and Communication, which captured from the park attendees' apps during one weekend (Friday, Saturday, and Sunday).

3.2.2 Movement Data

Table 3.1 Movement data

Timestamp	**Id**	**Type**	**X**	**Y**
06/06/14 08:00:11	1591741	check-in	63	99

The values are as follows; timestamp, person-id, type of activity (either check-in or movement), X is coordinate, and Y is coordinate. The park area is gridded to assist in specifying locations. The data file contains movement information around the grid, with coordinate locations. This data appears as in Table table 3.1. The movement data size is $103MB$. People either move from grid square to grid square or check in at rides, which means they either get in line or enter a ride. The information from the data as in table 3.1 for ID 1591741 show that a visitor checked into a ride located at $(63, 99)$ at 8:00:11 AM. So each visitor travels through the park their locations was recorded once they checked in . However , when there is no record during a particular second of time, it means that the individual has not moved out of their previous grid square but when visitor was not tracked after they check into a ride then they will eventually appear back on the grid when the ride is over.

3.2.3 Communications Data

Table 3.2 Communication data example

Timestamp	from	to	location	
2014-06-06 09:21:35.000	790661	1920255	Tundra Land	

This the second data file and contains communications information. The values are as follows: timestamp, from (the sender ID), to (the recipient ID), and location (the area where communications occurred). The location can be named as follows: Entry Corridor, Kiddie Land, Tundra Land, Wet Land, or Coaster Alley. The records are shown in table 3.2. The communications data size is 180MB. Graph 1 illustrates the difference in both data. The dimensions of the communications data include having four variables, with movement data being the fifth.

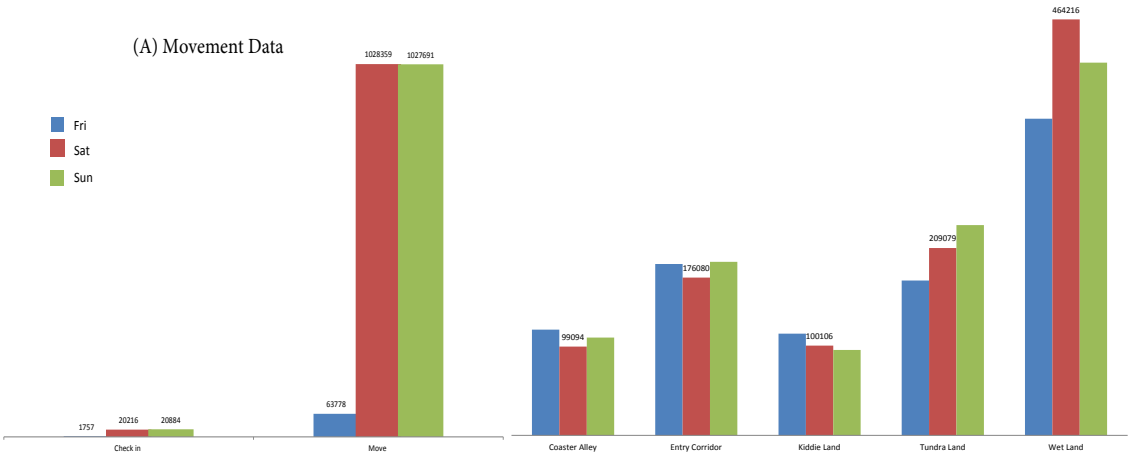


Figure 3.1 Information Recorded In Vast Dataset

3.2.4 Problems and Challenges

The scenario was in an amusement park. The simulated park covers a vast geographic space ($approx.500x500m^2$). Patterns can be found in the movement through the park and in the communications among visitors, including expected regular visit patterns and unexpected patterns.

Our tasks in the investigation were to focus on the movement of people around the park, how people move and communicate in the park, how patterns change and how the data change

and evolve over time. As we mentioned earlier, the dataset is large, and each data has a different size and values. Once the investigation is completed, the following questions should be answered: How big is the group type? Where does this type of group like to go in the park? How common is this kind of group? What are the other observations about this type of group? What can you infer about the group? If improvement in the park needed to be made to meet this group's needs, what would it be?

3.3 Experiments

The following steps taken into account before starting the test:

- Define the goal
- Gather and clean data
- Analyses and explore
- Choose technique and methods to extract and mining data
- Identify and Apply algorithm that we need to find pattern and predict result o Build model to finds pattern match and/or predicts result
- Evaluate results Iterate from explore and analyse the data (on-going feedback)
- Make decisions

3.3.1 Patterns Identifying Tasks

Cluster and Visualization Using R

Following graph 3.2 shows all the data before clustering, and visualizing the Friday Movement data in their three dimension, plotted it in 3D as follows:

The experimental tasks were time-consuming, especially with such a large set of data for identifying the patterns of large groups. The Friday movement data is visualised in three dimensions and plotted in 3D, as displayed in figure 3.2. R environment been used for both the data miners and the graphics. The original data must always be visualised in a graphical format to help make decisions about how many numbers should be to consider as K clusters. As we can see in Fig 3.3, the data point on left is the location of all the datasets after being plotted. This is the same as the real footpath in right figure for the park map data points.

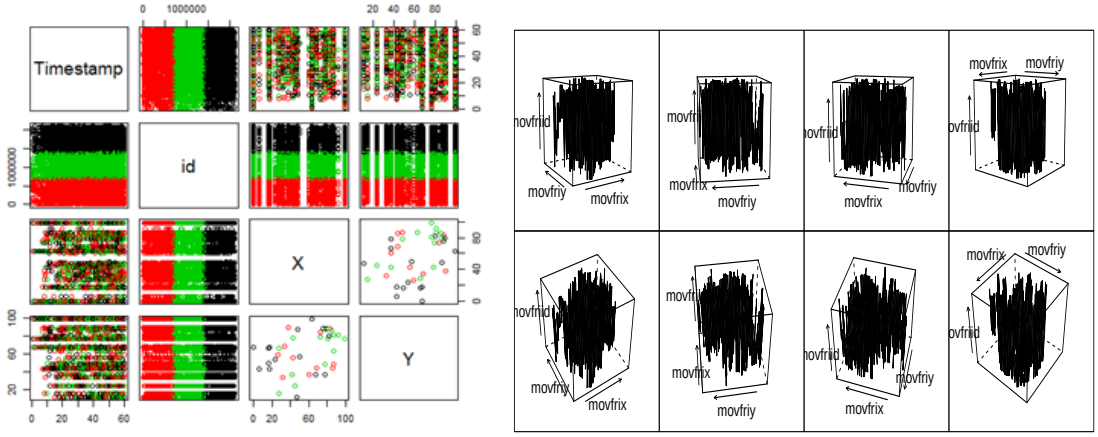


Figure 3.2 Visualization for all the dataset before clustering in 2D and 3D

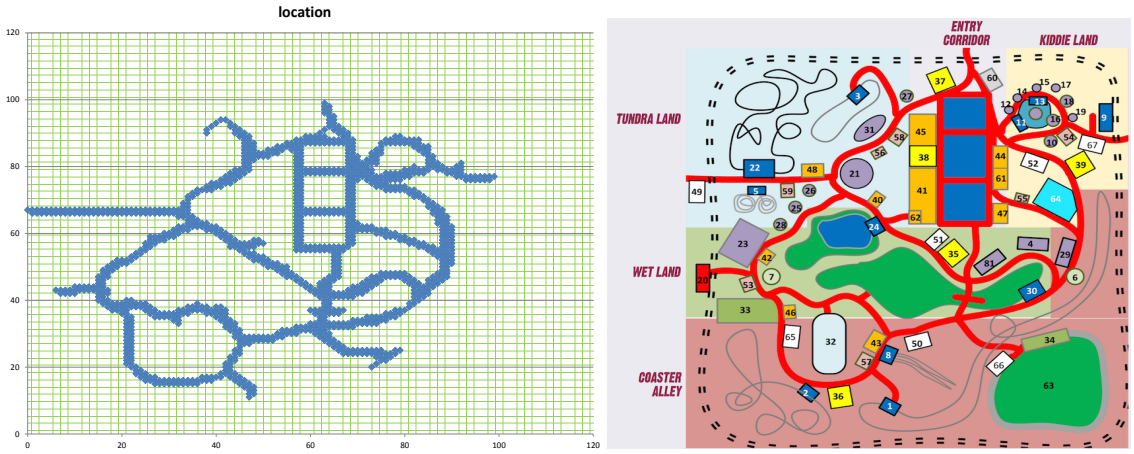


Figure 3.3 Location of all data on MATLAB same as The real Park map

3.3.2 Determining the Number of Clusters

For the clustering problem, we have been giving unlabelled data sets, and we would like to have an algorithm to group the data into subsets. In this clustering approach, the number of clusters needed to be decided in advance. The best way we found to determine the appropriate number of clusters was to plot the data as figure 3.4. From the dataset we have chosen Saturday and Sunday timestamps frequencies obtained by the SQL to compare data as figure 3.4.

Considering all the factors, in particular for the criminal data analysis, is a crucial aspect of making a decision that could lead us to a potential suspect.

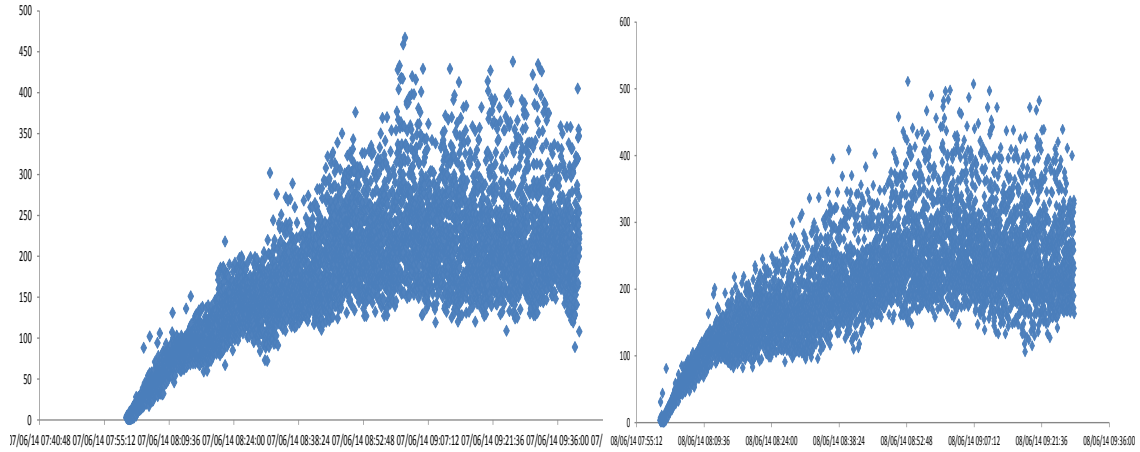


Figure 3.4 Determine the appropriate number of clusters

One factor at a time could still be used in the early stage of the analysing to assist later in determining how many clusters is need, but, determining how many clusters needed on only one factor it is inaccurate , because the ID and location as important values as the frequency of the timestamps . For example, in figure fig. 3.4, the plotting of the frequency of the Saturday and Sunday timestamps could easily help us to identify some patterns. The previous graph shows how many moves happened at a particular time on Saturday.

3.3.3 K-means Implementation

In our application, the first step in the loop of the K-means is the cluster assignment step, in which the cluster will go through each vector giving it a data set, and depending on whether it is closer or less close to the cluster centroid points, it will assign each data point to one of the cluster centroids.

The clustering algorithm goes through the data set, which records the time and id of each customer in the park. Then, when we plot them, we will be able to see each of the data points and how far or close they are to each centroid point they were assigned to. When we have decided that the number of the clustering $= 3$, then $K = 3$. Therefore, the X and Y are with the $3Cj$ centroid locations. The group we found can be seen in the following graph.

3.3.4 Modified K-means Implementation

We ran the algorithm for different numbers of k values, as in figure fig. 3.5, for $k = 3$ and $k = 5$. We then used the modified K-means algorithm based on the density. As we defined the best value of k as equalling 5, the space will be partitioned into $5 multiplied by 5$, which is

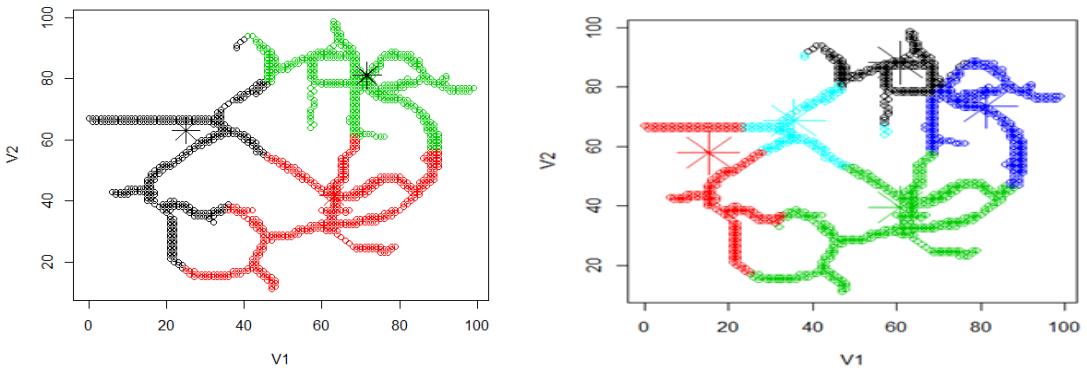


Figure 3.5 Clusters Obtained of k = 3 and k = 5

the same as the $k * k$ segments, as shown in figure 7. We then have to count the frequency of the data points in each section, as shown in the following Table.

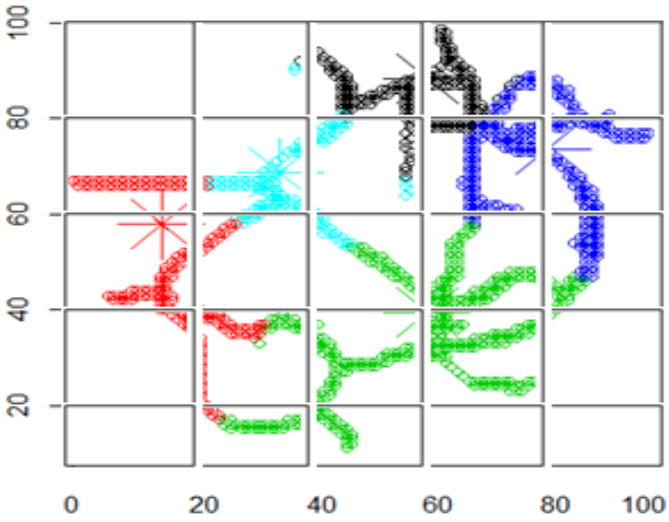


Figure 3.6 K-Means Results Partitioned into Different Segments

Then we have count the frequency of data point in each section, as shown in following table III .

There are many segments that show the highest frequency for a given k. Therefore, there is no merge for the segments due to the importance of each segment. Consequently, the centroids of the cluster are measured in all the segments. The centroid will be calculated using the mean of each segment. The cluster centroid then initialized, and we then assign each data point to each cluster's centroid by calculating the distance between each cluster's centroid and each data point. We then calculate the distance from any of the points from the centroid

Table 3.3 Number of data points in different segment

Segment	Number
((0,0),(20,20))	0
((0,20),(36, 67))	7280
((21,40),(15,93))	11074
((41,60),(20,40))	11842
((0,60),(20,80))	0
((61,80),(23,99))	24520
((0,80),(20,100))	0
((81,100),(38,88))	11032
((80,0),(100,20))	0

to compare them with what has calculated with each cluster's centroid and each data point. We then recalculate all the data points of each cluster individually. The clusters' centroids will change for each iteration. This phase must reiterate until the end of this condition is achieved.

3.4 Evaluation

K-Means was very important method for the data to be clustered, but it is not efficient enough to handle the criminal data analysis. Nevertheless, an improved version of this method was needed, and so on. The modified k-means algorithm approach used in our experiments which has improved the ability to overcome the limitations of the K-means algorithm, which was included the running of the algorithm for different k-value numbers and adjusting the initial point of the centroid location. By applying the modified k-means cluster, we have quickly identified how many K-means clusters needed to initialize the centroid point that lead to better clustering. But often modified k-means does not function well for high dimension, so an better version of this method was required, so we apply the PCA on original data set and obtain a reduced dataset containing possibly uncorrelated variables. PCA performs well when the data is disturbed by A random sample is a sequence of independent, identically distributed (IID) Guassian noise.

It means That PCA works well as long as the value of noise is not big enough. PCA it used by the feature transformation methods to help to reduce the data space to creates a new set of orthogonal variables that keep the same information as the original set, and then to find summarized set, but PCA dose not have as subspace clustering search methods and an evaluation criteria, and so the PCA used in feature transformation techniques does not help in

this instance, since relative distances are preserved and the effects of the irrelevant dimension remain.

3.5 Summary and Discussion

The Subspace Clustering of High Dimensional that we proposed to be use , is to interact with the input of the data based on the problems considered, and the algorithms will grouped data in way based on how they work. nonetheless, if the data is discomposed by arbitrary large noise, PCA will show a ineffective result, often even opposite to the possible solution, and that why author effective methods is needed, where will propose in the next chapter.

Chapter 4

Low Rank Sparse Representation for Subspace Clustering

4.1 New Affinity Metric for Low Rank Representation

To construct an affinity matrix offering more discriminative information for clustering, Liu et al. [83]. provide a post-processing step of low-rank representation **LRR** for subspace segmentation, and the LRR used to recover the subspace structures from the data that has errors, and (LRR) used [86]. Given a set of data samples each of which can be represented as a linear combination of the bases in a dictionary, LRR aims at finding the lowest-rank representation of entire data together. The computational procedure of LRR is to solve a nuclear norm [139] regularized optimization problem, which is convex and can be solved in polynomial time. By choosing a specific dictionary, it is shown that LRR can well solve the subspace clustering problem: when the data is clean, LRR prove exactly recovers the row space of the data; for the data contaminated by outliers . Their experiments confirm a significant improvement over clustering performance. In the following section, we will give a further analysis and formally formulate it as a similarity measure criterion and name it as LRR-COS.

Denote $\mathbf{L} = \mathbf{AZ}$ the clean data recovered by LRR and its **SVD** $L = U_r S_r V_r^T \in \mathbb{R}^{M \times N}$, rank($\mathbf{L}$) = r . In fact, the column vectors of $\mathbf{U}r$ denotes basis vectors spanning the column space of $\mathbf{L}$ and the corresponding singular value of $\mathbf{S}r$ specifies how important a basis vector is.

Column vectors in $\mathbf{V}_r^T \in \mathbb{R}^{r \times N}$ are new representation of the data $\mathbf{L}$ under the scaled and rotated coordinate system $\mathbf{U}_r\mathbf{S}_r$. The low-rank representation $\mathbf{Z} = \mathbf{V}_r\mathbf{V}_r^T$ is actually the pairwise similarity metric defined by inner product: $[\mathbf{Z}]_{ij} = \mathbf{v}_i^T\mathbf{v}_j, \mathbf{v}_i = [\mathbf{V}_r^T]_{:,i}\,\mathbf{v}_j = [\mathbf{V}_r^T]_{:,j}$. It is well-known that the similarity metric defined by cosines between representation vectors is more reasonable than using the inner product especially when subspaces are disjoint or not strictly independent. Therefore, we define a new affinity matrix for the graph as follows:

$$\Big[W\Big]_ij \quad = \quad \left(\frac{v_i^T v_j}{\| v_i \|_2 \| v_j \|_2}\right)^2, \mathbf{v}_i \quad = \quad [\mathbf{V}_r^T]_{:,i}, \mathbf{v}_j \quad = \quad [\mathbf{V}_r^T]_{:,j}. \tag{4.1}$$

Using squared cosine values $(.)^2$ in equation 4.1 is to ensure that the values of the affinity matrix W are positive.

In the ideal noise-free cases, when we take **SVD Z** $=$ $\mathbf{U}_z \sum_z \mathbf{V}_z^T$, we can infer that $U_z = V_z \quad = \quad V_r$ and $\sum_z$ is the identity matrix. In general, we deduce that $U_z(\sum_z)^{1/2} \approx V_r$. Then given the coefficient matrix $\mathbf{Z}$ obtained by LRR, we construct the affinity matrix $\mathbf{W}$ as follows:

$$\Big[W\Big]_ij \quad = \quad \left(\frac{u_i^T u_j}{\| u_i \|_2 \| u_j \|_2}\right)^2, u_i \quad = \quad [\mathbf{U}_W]_{:,i}, \mathbf{u}_j \quad = \quad [\mathbf{U}_W]_{:,j}. \tag{4.2}$$

where $\mathbf{U}_W \quad = \quad \mathbf{U}_z(\sum_z)^{1/2}$.

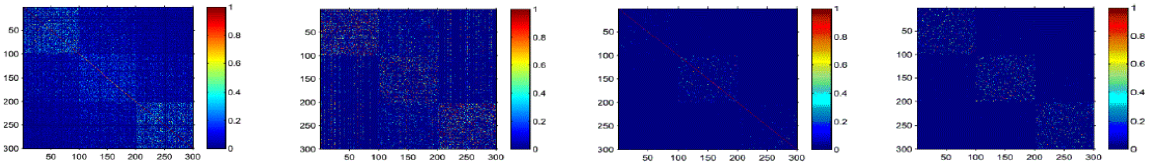
4.2 LRSR and Optimization Algorithms

In this section, we first propose Low Rank subspace Sparse Representation (LRSR) model for subspace clustering, then propose an efficient scheme to solve the proposed LRSR model via the linearized ADM [140] framework.

4.2.1 The LRSR model

As analysed above, low rank representation can capture global information critical for revealing low rank subspace structure and can remove large disturbance in original data. LRR has excellent performance in regard to analysing corrupted data drawn from independent subspaces. However, using the original data contaminated with large noises as the dictionary

is by no means a nice choice. Moreover, **LRR** often fails in the case of disjoint subspaces or overlapping subspaces.



(a) LRR (b) SSC (c) LRR-COS (d) LRSR

Figure 4.1 Disjoint subspace data X with ambient dimension

Figure fig. 4.1 : Disjoint subspace data X with ambient dimension, subspace dimension and subspace angle $(1000, 100, 42.3)$: Percentage of outliers 0.005 $||X||_F^2$ is 16.7% Images above are the results of **LRR**, **SSC**, **LRR-COS** and **LRSR** from the left to the right with the SG Accuracy 48.79%, 82.35%, 62.56%, 86.08% respectively.

For example, in the data used in the experiment shown in fig. 4.1, the first and the third subspace are independent but the second subspace has intersections with each of the others. Then, we get a coefficient matrix that is far from the ideal *Block-Diagonal* one. Further, to achieve a strict *Block-Diagonal* coefficient matrix without post-processing step, **LRR** is only applied to data from orthogonal subspaces. **SSC** is superior to **LRR** in this regard. However, it is weaker in recovering the global subspace structure from the corrupted data than **LRR**. The result of **SSC** in Figure fig. 4.1 shows that **SSC** does not remove the impact of large noises.

Motivated by the above observation and analysis, we propose a method to combine the low rank and sparse representation for subspace clustering especially for the cases when the subspaces are not independent and data are corrupted by large noises. For example, the corruptions caused by the uneven illumination leads to a relatively large number of within cluster data points spread to other subspaces. Therefore, local methods like SSC may bring about points from different subspaces into the same subspace with the uneven illumination corruption. Nevertheless, LRR is better at handling corruptions than SSC. The collection of face images of multiple subjects lie close to a union of several 9-dimensional subspaces [141].

However, we cannot ensure the face image subspaces are orthogonal or independent to each other. In other words, LRR recovers an affinity matrix with dense between-class affinities. Therefore, we extend the framework by learning a clean dictionary (bases for subspaces) which satisfies the convergence condition of SSC, i.e., favouring relatively smaller within-

subspace distances and larger principal angle between subspaces. The model proposed is defined as follows:

$$\begin{aligned} \hat{\mathscr{J}} = LRSR \quad \min_{Z,A,J,E} \quad & ||A||_* + ||Z||_* + \lambda_2||J||_1 + \lambda_1 ||E||_{2,1}, \\ s.t. \quad & X = AZ+E, \quad Z = J - diag(j). \end{aligned} \tag{4.3}$$

Different from SSC and LRR models, the proposed model (4.3) aims to recover both a clean dictionary A and a low-rank-sparse coefficient matrix Z simultaneously. E represents the sample-specific corruptions. As rank $(A^*Z^*) \leq \min\{rank(A^*), rank(Z^*)\}$, A^*Z^* is the low rank recovery of the original data. The optimization problem (4.3) can be solved in an alternative iteration fashion, in which we need to solve two following sub-problems**:**

1 Fixing Z and J, solve (4.3) for A and E by the following problem, denoted by LRSR-1,

$$\begin{aligned} \min_{A,E} \quad & ||A||_* + \lambda_1||E||_{2,1} \\ s.t. \quad & X = AZ + E. \end{aligned} \tag{4.4}$$

2 Fixing A and E, solve (4.3) for Z and J by the following problem, denoted by LRSR-2,

$$\begin{aligned} \min_{Z,J} \quad & ||Z||_* + \lambda_2||J||_1 \\ s.t. \quad & X = AZ, \\ Z = J - & diag(j). \end{aligned} \tag{4.5}$$

Sub-problem (LRSR-1) can be considered as a subspace segmentation extension for RPCA. As discussed in [142], RPCA cannot handle well mixed data, since it assumes the whole dataset is drawn from a single subspace. LRSR-1 is a general one that can leverage the influence of both RPCA and sparse-coding subspace segmentation for data sampled from a union of subspaces. Based on the analysis in the previous subsections, we now propose to solve each sub-problem by the linearized ADM method [140].

4.2.2 Alternating scheme for LRSR

We will develop an alternating scheme for solving both LRSR-1 and LRSR-2,respectively. Both LRSR-1 and LRSR-2 fall in the ADMM framework or its extension [92], it is straightforward to formulate an iterative algorithm for them. Let us start with LRSR-1 by using the augmented Lagrangian multiplier method. The augmented Lagrangian function of problem LRSR-1 is defined as

$$\begin{aligned}\zeta 1 \quad (A,E,Y_1) = \quad & ||A||_* + \lambda_1||E||_{2,1} + \left\langle AZ+E-X, Y_1 \right\rangle \\ & + \tfrac{\mu}{2} \parallel AZ+E-X \parallel_F^2 ,\end{aligned}$$

where $Y_1 \in \mathbb{R}^{M \times N}$ is the Lagrangian multiplier and $\mu < 0$ is a penalty parameter to be adaptively updated. Then the Alternating Direction Method (ADM) [92] for minimizing $\zeta 1$ can be written into the following iteration procedure:

$$A^{k+1} = ||A||_* + \frac{\mu_k}{2}||AZ+E^k-X+Y_1^k/\mu_k||_F^2, \tag{4.6}$$

$$E^{k+1} = \lambda_1||E||_{2,1} + \frac{\mu_k}{2}||A^{k+1}+Z+E-X+Y_1^k/\mu_k||_F^2, \tag{4.7}$$

$$Y_1^{k+1} = Y^{k+1} + \mu_K(A^{k+1}Z+E^{k+1}-X), \tag{4.8}$$

$$\mu + 1 = \min(\rho\mu_k, \mu_{max}), \tag{4.9}$$

Both ρ and μ max in (4.9) are tunable parameters controlling the convergence of the algorithm.

ADM is very efficient, provided that the resulting sub-problems (4.6) and (4.7) have closed-form solutions. In fact, according to [[143], lemma 3.3] , a closed-form solution to (4.7) is given as follows:

$$\begin{aligned}[E^{k+1}]_{:,i} \quad & = max\left\{||h_i||_2 - \frac{\lambda_1}{\mu k}, 0\right\}\frac{h_i}{||h_i||_2}, \\ & where \quad H = X - A^{k+1}Z - \frac{Y_1^k}{\mu k}\end{aligned} \tag{4.10}$$

where h_i is the $i-th$ column vector of $\mathbf{H}$.

Although (4.7) has a closed-form solution as defined in (4.10), there is no known closed-form solution to (4.6) because of the involving coefficient Z. Fortunately, the linearized ADM method [140] has been proposed to solve 4.6 efficiently without introducing auxiliary variables.

In (4.6) , denote $F(A) = \| AZ + E^k - X + Y_1^k/\mu_k \|_F^2$. Taking the linearized approximation of $F(A)$ at the current A^k, we have

$$F(A) \approx F(A^k) + \left\langle \nabla F(A^k), A - A^k \right\rangle + \frac{\tau A}{2} \| A - A^k \|_F^2$$

where $\tau A > \rho(Z^T Z)$ is the proximal parameter, $\rho(Z^T Z)$ denotes the spectral radius of $Z^T Z$, and $\nabla F(A^k)$ denotes the gradient of the function $F(A)$ at A^k. (4.6) can be reformulated by linearized ADM as follows:

$$\begin{aligned} A^{k+1} &= arg\min_A ||A||_* + \frac{\mu k \tau A}{2} ||A - A^k + Y_A/(\mu k \tau A)||_F^2, \\ &\text{Where } Y_a = (Y_1^K + \mu_k (A^k Z + E^k - X)) Z^T \ . \end{aligned} \tag{4.11}$$

Problem (4.11) has a closed-form solution given by the thresholding operator, see [[93] , Theorem 2.1],

$$\begin{aligned} A^{K+1} &= U \delta_1 / \mu K \tau A (\Sigma) V^T, where\ \mathbf{A}^K - Y_A(\mu k \tau A) = U \Sigma V^T, \\ \delta \frac{1}{\mu k \tau A}(\Sigma) &= max(\Sigma - \frac{1}{\mu k \tau A}, 0) + min(\Sigma - \frac{1}{\mu k \tau A}, 0) \end{aligned} \tag{4.12}$$

Now we consider the subproblem LRSR-2, which it is reasonable to be reformulated by introducing an auxiliary variable $\mathbf{Q}$ as follows:

$$\begin{aligned} &\min_{Z,Q,J} ||Q||_* \quad + \quad \lambda_2 ||J||_1 \\ s.t. \quad &X - E = AZ, \quad Q = Z, \quad Z = J - diag(J). \end{aligned} \tag{4.13}$$

The corresponding augmented Lagrangian function of (4.13) is

$$\begin{aligned} \zeta_2(Z,Q,J) \quad &= ||Q||_* + \lambda_1 ||J||_1 \quad + \quad \frac{\beta k}{2} ||X - E - AZ + Y_2/\beta_k||_F^2. \\ &+ \frac{\beta k}{2} ||Z - J + diag(J) + Y_3/\beta_k||_F^2 + \frac{\beta k}{2} ||Z - Q + Y_4/\beta_k||_F^2. \end{aligned} \tag{4.14}$$

Similar to deriving the iterative procedure for problem LRSR-1, we can propose the following iterative updating scheme for minimizing $\zeta 2$ (Z,Q,J):

$$Z^{k+1} = (A^T A + 2I)^{-1}(J^k + A^T(X - E + \frac{1}{\mu_K} Y_2^K) - \frac{1}{\beta_k}(Y_3^k + Y_4^k)), \tag{4.15}$$

$$Q^{k+1} = U\delta_1/\beta_k(\Sigma)V^T, \quad where \quad Z^k + Y_4^K/\beta_k = \quad U\Sigma V^T, \tag{4.16}$$

$$J^{k+1} = \quad T - diag(T), \quad whereT = \delta_1(Z^k + Y_3^k/\beta_k), \tag{4.17}$$

$$Y_2^{k+1} = \quad Y_2^k + \beta_k(X - E - AZ^k), \tag{4.18}$$

$$Y_3^{k+1} = \quad Y_3^k + \beta_k(Z^k - J^k), \tag{4.19}$$

$$Y_4^{k+1} = \quad Y_4^k + \beta_k(Z^k - Q^k), \tag{4.20}$$

$$\beta k + 1 = \min(\rho\beta_k, \beta_{max}), \tag{4.21}$$

4.3 Stopping Criteria for LRSR-1 and LRSR-2

The KKT conditions optimization, is the Karush–Kuhn–Tucker (KKT) conditions that are derivative tests necessary for a solution in non-linear programming to be optimal, provided regularity conditions are satisfied. KKT conditions of LRSR-1 are that there exists a triple (A^*, E^*, Y_1^*) such that

$$A^* Z + E^* - X = 0\,,$$

$$Y_1^* Z^T \in \partial||A||_*, \quad Y_1^* \in \partial||E||_{2,1}$$

The triple (A^*, E^*, Y_1^*) is called a KKT point. So the first stopping criterion is the feasibility:

$$||A^{k+1}Z + \quad E^{k+1} - X||_F/||X||_F \quad < \varepsilon_1 \tag{4.22}$$

Furthermore, (A^{k+1} ,E^{k+1} , ${Y_1}^{k+1}$) should satisfy the second KKT condition as follows:

$$\begin{aligned} -\mu_k \tau_A (A^{k+1} - A^k) Z^T \quad + Y_1^{k+1} Z^T \in \partial ||A||_*, \\ -\mu_k [E^{k+1} - E^k \quad + (A^{k+1} - A^k) Z^T] \quad + Y_1^{k+1} \quad \in \lambda_1 \partial ||E||_{2,1}, \end{aligned} \tag{4.23}$$

For ${Y_1}^{k+1}$ to satisfy the KKT conditions of the problem in (4.23), both $\mu_k \tau A \parallel A^{k+1} - A \parallel_F^2$ and $\mu_k \parallel E^{k+1}$ - $E^k \parallel_F^2$ should be small enough. Hence, we have the second stopping criterion for LRSR-1:

$$\begin{aligned} max(||E^{k+1} - \quad E^k||_F / ||X||_F, \\ \mu_k \tau_A ||A^{k+1} - A^k||_F / ||XZ^T||_F < \varepsilon_2) \end{aligned} \tag{4.24}$$

The stopping criteria of LRSR-2 can be obtained in the similar way as follows:

$$\begin{aligned} max(||X - E - AZ^{k+1}||_F / ||X||_F, \\ ||Z^{k+1} - J^{k+1} + diag(J^{k+1})||_F / ||Z^{k+1}||_F) < \varepsilon_1 \end{aligned} \tag{4.25}$$

$$- \quad \mu_k ||Z^{k+1} - Z^k||_F / ||A^T X \quad - \quad E||_F < \varepsilon_2 \tag{4.26}$$

Finally we summarize the iterative scheme for solving eq. (4.3) in Algorithm 1 and the general framework of LRSR for subspace clustering in Algorithm 2 in the following Experiments chapter.

4.4 Experiments on Hopkins dataset

HOPKINS 155 DATASET from vision lab has been created with the goal of providing an extensive benchmark for testing feature based motion segmentation algorithms. It contains video sequences along with the features extracted and tracked in all the frames. The ground-truth segmentation is also provided for comparison purposes. The data is stored.

4.4.1 Experiments with Synthetic Data

Three disjoint subspaces been consider $\{S_i\}_{i\ =\ 1}^{3}$ of the same dimension d embedded in the $D-$ dimensional ambient space, where both D and d are to be specified below. We further assume that S_1 is orthogonal to S_3, and S_2 is to be constructed specifically below.

Denote by $\mathbf{U}_i \in \mathbb{R}^{D\ \times\ d}$ the orthogonal basis of $\mathbf{S}_i$ $(i = 1$ and $3)$. we make sure that rank $([\ \mathbf{U}_1\ \mathbf{U}_3]\ ) = 2_d$, rank $([\ \mathbf{U}_2\ \mathbf{U}_3]\ ) = 2_d$, rank $([\ \mathbf{U}_1\ \mathbf{U}_2]\ ) = 2_d$, but rank $([\ \mathbf{U}_1\ \mathbf{U}_2\ \mathbf{U}_3]\ ) \neq 3d$ to satisfy the condition of *disjoint subspaces*. Then we constructed $\mathbf{U}_2$ as a linear combination of $\mathbf{U}_1$ and $\mathbf{U}_3$.

Then we randomly divide both $\mathbf{U}_1$ and $\mathbf{U}_3$, individually, into 2 parts (50 % each) denoted by $\mathbf{U}_1 \triangleq [\ \mathbf{U}_{11}\ \mathbf{U}_{12}]$ and $\mathbf{U}_3 \triangleq [\ \mathbf{U}_{31}\mathbf{U}_{32}]$.

Then we defined $\mathbf{U}_2 = [\ \mathbf{U}_{21}\mathbf{U}_{22}]$ in the same size as $\mathbf{U}_1$, where the columns in $\mathbf{U}_{21}$ are linearly combined by $[\ \mathbf{U}_{11}\mathbf{U}_{31}]$ and the columns in $\mathbf{U}_{22}$ are linearly combined by $[\ \mathbf{U}_{12}\mathbf{U}_{32}]$ at random.

Then we randomly generated the same number of data points $n_d = 100$ in each subspace.

In our experiment, we set $D = 1000$ and $d = 36, 48, 72, 84, 120$, respectively.

The matrix $\mathbf{X} = [\ X_1, X_2, X_3]$, is formed by the samples generated from disjoint subspaces S_1, S_2, S_3 respectively, with larger Laplacian noises of mean 0:005 $\|X\|_F$ added, and the amount of the Laplacian noises at 16:7 % .

Each data point x_i coming from S_{ki} ($\mathrm{k_i} = 1,$ or 2 or 3) , denote its new representation by ${z^T}_i$ $z_i^T \triangleq [\ {z_{i1}}^T, {z_{i2}}^T, {z_{i3}}^T]$, where z_{ij} denotes the representation coefficients corresponding to the points in S_j . We measure the subspace segmentation accuracy by the following *SG* criterion :

$$\text{SG Accurancy } = \frac{1}{3n_g}\sum_{i=1}^{3n_d}\frac{\|z_{ik_i}\|_2}{\|z_i\|_2} \in [\ 0,1]\ , k_{\mathrm{i}}\ = \{1,2,3\}. \tag{4.27}$$

where each term inside the summation indicates the fraction of the l_2-norm of z_i over the same subspaces. However, because of the unknown dictionary $\mathbf{A}$ in **LRSR**, we should ensure the column vector a_i, $(i = 1 \ldots m)$ in A correspond to which subspace as follows:

$$\mathrm{k_i} = \arg\max_i\ \|\ X_j^T a_i\ \|_F^2\ /\ \|\ X_j a_i\ \|_F^2\ (j = 1\ ,2\ ,3\ ) \tag{4.28}$$

Then we can use (4.11) to evaluate the subspace segmentation performance of LRSR. In the synthetic experiment, we set $\lambda = 1.2$ for SSC, $\lambda = 9$ for LRR but this parameter has tiny influence on the result and $\{\lambda_1 = 0:40, \lambda_2 = 0.73\}$ for LRSR.

In Figure fig. 4.1, we demonstrate the coefficient matrix Z of LRR, SSC, LRR-COS and LRSR. As we can see from the figure, LRR leads to non-zero coefficients among subspaces $\{S_2, S_3\}$ and subspaces $\{S_2, S_1\}$. As to the SSC sub-figure, the noises impact can be explicitly observed. LRR-COS can handle both the segmentation and the large corruptions better than LRR but lack within-class connectivity. Obviously, the coefficient matrix given by LRSR has a relatively ideal diagonal block structure. What's more, we also show the infuence of the subspace dimension to segmentation accuracy in Figure fig. 4.2.

We can conclude that LRSR has the best result and LRR-COS behaves better than LRR. Rather than simply measuring the clustering error by computing the percentage of the misclassified points in total points [7], [86], we introduce an evaluation metric, the F-measure [144], involving both of precision rate and recall rate for evaluating the clustering performance, defined in the following way: First calculate $\mathrm{F}_(i)$ for the i-th manual annotation subset as

$$F(i) = \max_j \frac{2 * \text{precision}(P_i, C_j) * \text{recall}(i, j)}{\text{precision}(P_i, C_j) + \text{recall}(P_i, C_j)}, \quad (i = 1, \cdots, n_p, j = 1, \cdots, n_c) \tag{4.29}$$

where P_i and C_j denote the i-th manual annotation subset and the j-th clus340 tering subset respectively. n_p and n_c are the numbers of them. Then the F-measure, the global evaluation metric for clustering, is defined as follows:

$$F\text{-measure} = \sum_{i=1}^{n_p} \frac{n_i}{n_p} F(i), \tag{4.30}$$

where n_i is the number of manual annotation set P_i.

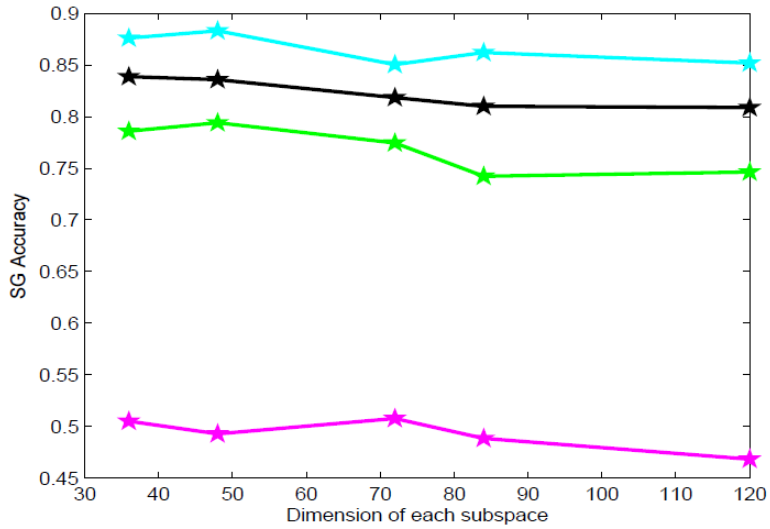


Figure 4.2 of LRR, LRR-COS, SSC and LRSR on the synthetic data of three disjoint subspaces with ambient dimension, subspace angel, large noises mean (1000,44.5,0:005), while x-axis represents the intrinsic subspace dimension.

4.4.2 Clustering

Given face images of multiple subjects with a fixed pose and varying illumination, we consider the clustering problem according to subjects. It has been shown that, under the Lambertian assumption, images of a subject with a fixed 345 pose and varying illumination lie close to a linear subspace of dimension 9 [141]. Thus, the collection of face images of multiple subjects lie close to a union of several 9-dimensional subspaces. In reality, the errors correspond to the cast shadows and specularities in the face images and can be modelled as sparsely outlying entries.

Figure 4.3 given face images of di erent subjects shoot under di erent light- ing conditions (top), our goal is to cluster images that belong to the same subject (bottom).

In this section, we evaluate the clustering performance of **LRSR** and **LRR-COS** as well as the state-of-the-art methods on the Extended Yale **B** dataset. For example, Figure fig. 4.3 shows face images of five subjects from Extended Yale **B**.

The dataset consists of 192×168 pixel cropped face images of $n = 38$ individuals, where there are $N_i = 64$ frontal face images for each subject acquired under various lighting conditions. We down sample the images to 48×42 pixels and treat each 2016-dimensional vectorized image as a data point. The face images are corrupted by errors such as shades or highlight.

Face[2]	**5 Subjects**	**5 Subjects**	**10 Subjects**	**15 Subjects**	**20 Subjects**
SSC	0.712±.100	0.724±.094	0.680±.148	0.595±.186	0.609±.154
SSC-N	0.821±.079	0.940±.047	0.816±.119	0.830±.108	0.802±.127
LRR	0.783±.08	0.754±.150	0.696±.173	0.723±.163	0.697±.160
LRR-COS	0.853±.08	0.951±.048	0.843±.117	0.821±.174	0.816±.156
LRSR	0.878±.068	0.941±.052	0.909±.089	0.867±.111	0.877±.182

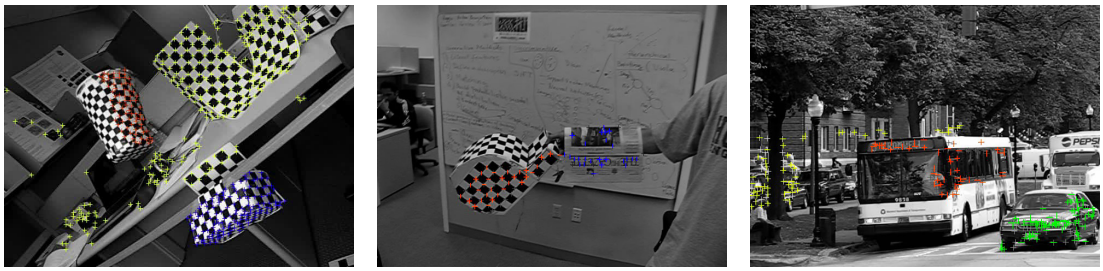
Table 4.1 F-measures of different methods on the Extended YaleB database

In this experiments, we trial on five groups of n subjects, $n \in \{58101520\}$. We use F-measure (eq. (4.30)) as the evaluation metric. Table table 4.1 shows the F-measure of different algorithms for diffierent groups of subjects. In this experiments, we set $\lambda = 0.12$ for LRR,$\lambda = 1.5$ for SSC and $\lambda\ 1 = 0.6$, $\lambda 2 = 0.01$ for LRSR.

Without post-processing its coefficient matrix, LRR has the lowest *F-measure* among all the algorithms. The post-processing method (LRR-COS) significantly improves the clustering performance. It recovers the row space of original face images. SSC also has dis-satisfactory performance because of the cast shadows and specularities in face images. SSC seeks the locality information of each sample. For face images, the neighbourhood of each data point contains points that belong 370 to other subjects and, in addition, the number of neighbours from other subjects increases when we increases the number of subjects. LRSR has the best performance among all the algorithms. In the case of 20 subjects, LRSR outperforms the LRR-COS by about 8 percent in F-measures.

4.4.3 Segmentation

Motion segmentation refers to the problem of separating a video sequence into multiple spatio-temporal regions corresponding to different rigid-body motions in the scene. The motion segmentation problem can be preceded by first extracting a set of feature points $\{X_{fi} \in \mathbb{R}^2\}_{i=1}^{N}$ from the video sequences $f = 1, ..., F$ using standard tracking methods. Each



(a) A checkerboard (b) An arm (c) A cars10

Figure 4.4 Example image frames of three categories motion sequences Checkerboard, Articulated and Traffic from the Hopskins 155 database Marked Featured points in each sub-fgure with the same colour correspond to the same motion.

data point yi, which is also called a feature trajectory, corresponds to a $2F$-dimensional vector obtained by stacking the feature points $\{X_{fi}$ in the video as

$$\mathbf{y}_i \triangleq \begin{bmatrix} \mathbf{x}_{1i}^T & \mathbf{x}_{2i}^T & \cdots & \mathbf{x}_{Fi}^T \end{bmatrix}^T \in \mathbb{R}^{2F}. \tag{4.31}$$

Then the problem reduces to clustering these points trajectories according to different rigid-body motions. Under the affne projection model, all feature trajectories associated with a single rigid motion lie in a linear subspace of dimension at most 4 in $\mathbb{R}^{2F}$ [145], [146]. Hence, motion segmentation reduces to clustering of data points in a union of subspaces.

In this subsection, we apply LRSR to motion segmentation problem and evaluate it on the Hopkins155 motion database, which is available on-line at

Table 4.2 F measures with 2 motions

2 motions	**Checkerboard**	**Articulated**	**Traffic**
SSC	0.731 0.17	0.953 0.031	0.961 0.102
LRR	0.821 0.207	0.90 0.129	0.989 0.307
LRSR	0.878 0.068	0.941 0.118	0.967 0.216

Table table 4.2 is the F-measures of different methods on the Hopkins155 with 2 motions respectively, where λ 1 = 0.2, 2 = 0.01 for LRSR

Table 4.3 F measures with 3 motions

2 motions	**Checkerboard**	**Articulated**	**Traffic**
SSC	0.701±0.145	0.931±0.174	0.940±0.095
LRR	0.815±0.256	0.883±0.202	0.964±0.223
LRSR	0.878±0.213	0.962±0.161	0.967±0.107

Table table 4.3 is F-measures of different methods on the Hopkins155 with 3 motions respectively, where λ 1 = 0.2, 2 = 0.01 for LRSR

Table 4.4 F-measures of different methods on the four Hopkins sequences with outliers and missing data

Hopkinsadd	**books**	**carsturning**	**carsbus**	**nrbooks3**
SSC	**0.278**	**0.278**	**0.278**	**0.278**
SSCN	**0.278**	**0.278**	**0.278**	**0.278**
LRR	**0.278**	**0.278**	**0.278**	**0.278**
LRSR	**0.278**	**0.278**	**0.278**	**0.278**

4.5 Evaluation

In this section, we evaluate the performance of our proposed LRSR applied to big and Criminal Data Analysis comparing to algorithms which has applied on both synthetic data. we proved of a subspace clustering framework called LRSR Applied to Criminal Data Analysis is better than the PCA segmented and K-means, as it obtains the optimal solution based on both LRR and sparse representation. Our proposed application can successfully reveal multiple subspace structures and to recover the low-dimensional subspaces and to seek a low-rank and sparse representation for clustering. However, there still remain several problems with the current LRSR model. Our future work will includes the investigation of a systematic way to estimate the parameters. In addition, we are also interested in extending the current model from unsupervised clustering to semi-supervised clustering that is capable of involving prior knowledge.

We first introduce two theorems regarding the convergence of the augmented Lagrangian multiplier algorithms for sub-problems LRSR-1 and LRSR-2.

Sub-problem LRSR-1 is similar to the transposed standard LRR model, thus the convergence analysis in [140] can be applied to this model. We present a convergence theorem below.

Theorem 1 *If* $\{\mu_k\}$ *is non-decreasing and upper bounded,* $\tau_A > \rho\,(ZZ^T)$*, then the sequence* $A^k, E^k, Y_1{}^k$ *generated by and eq.* (4.6) *eq.* (4.9) *converges to a KKT point of LRSR-1.*

For the revised LRSR-2 model eq. (4.13), there are three blocks of primary variables. For the cases of more than two blocks of primary variables, a naive linearized version of ADM may not converge. As suggested in [147], a parallel version has been adopted in eq. (4.15) to eq. (4.21) . From the convergence analysis in [147], we immediately have the following theorem.

Theorem 2 *If*$\{\beta_k\}$ *is non-decreasing and upper bounded, then the sequence* $(Z^k, J^k, Y_2{}^k, Y_3{}^k, Y_4{}^k)$ *generated by eq.* (4.15) *- eq.* (4.21) *converges to a KKT point of LRSR-2.*

Proof Please refer to the proof in [147].

Theorem 3 *Let the sequence of LRSR-1* $\{(A^k, E^k)\}$ *converge to* $(A_j{}^*,\ E_j{}^*)$ *and that of LRSR-2* $\{(Z^k, J^k)\}$ *to* $(Z_j{}^*,\ J_j{}^*)$ *we can prove that sequence of* $\hat{\mathscr{J}}$ *LRSR* $(A_j{}^*,\ E_j{}^*,\ Z_j{}^*,\ J_j{}^*)$ *decreases and a local minimal can be obtained.*

Proof According to the models of LRSR-1 and LRSR-2 in eq. (4.4) and eq. (4.4), respectively, we can obtain that

$$\begin{aligned} \hat{\mathscr{J}}\,LRSR \quad (A_J^*, E_J^*, Z_J^*, J_J^*) \quad &\geq \hat{\mathscr{J}}\,LRSR \quad (A_{J+1}^*, E_{J+1}^*, Z_J^*, J_J^*) \\ &\geq \quad \hat{\mathscr{J}}\,LRSR \quad (A_{J+1}^*, E_{J+1}^*, Z_{J+1}^*, J_{J+1}^*) \end{aligned} \tag{4.32}$$

The objective function decreases at each alternating minimization step. Therefore, the sequences $\hat{\mathscr{J}}$ LRSR $(A_t,\, E_t,\, Z_t,\, J_t)$, $(t = 1, 2, \ldots)$ is decreasing. From both Theorems theorem 1 and theorem 2, the sequence $(\{A_t,\, E_t,\, Z_t,\, J_t\})$ is bounded, so, without loss of generality, we have an accumulation point $(A_\infty,\, E_\infty,\, Z_\infty,\, J_\infty)$ such that it is local minimum of $\hat{\mathscr{J}}$ LRSR.

4.6 Summary and Discussion

Subspace clustering framework called LRSR Applied to Criminal Data Analysis, obtains the optimal solution based on both low rank representation (least number of dimensions) and sparse representation. Low rank subspace recovery detected corruptions and limiting the self-expressive coefficient matrix to be sparse for disjoint subspaces, and algorithm successfully reveal multiple subspace structures. LRSR algorithm aims to optimize nuclear norm of the dictionary to recover the low-dimensional subspaces and to seek a low-rank and sparse representation for clustering. Numerical experiments have confirmed the comparable/superior performance of the methods over for both synthetic data and real application data (Criminal Data). As shown in Figure fig. 4.4, the database consists of 155 sequences of two and three motions which can be divided into three main categories: *Checkerboard*, *Articulated* and *Traffic* sequences.

The trajectories are extracted automatically by a tracker and outliers are manually removed. Therefore, the trajectories are only corrupted by noises without any missing entries or outliers. We perform the model learning with the original 2F-*dimensional* feature trajectories and utilize the metric of F-measure to evaluate the performance of the methods. The average and median F-*measures* are listed in Table table 4.2 and table 4.3. In order to compare LRSR with the state-of-the-arts, we also list the results of LRR, SSC and SSC-N. SSC-N has a post-processing step by only retaining the k + 1 -largest elements in each column of the coefficient matrix, i.e., keeping the $k+1$ -nearest neighbours of each sample. Without the

post-processing, SSC has the highest error among all the algorithms. On the other hand, the post-processing over the sparse-representation matrix by only reserving the k + 1 -largest coefficient in each column significantly improves the performance (SSC-N). In Table Table table 4.2 and table 4.3, we show the results of LRR, SSC and LRSR evaluated on the Hopkins database of 2 motions and 3 motions sequences. As the motion sequences were obtained using an automatic tracker, and errors in tracking were manually corrected for each sequence, it is reasonable to believe that these sequences only contain small noises. In fact there has been very few attempts to deal with heavily corrupted trajectories. All the methods perform well for video sequences without missing data and outliers. In Table table 4.4, we also demonstrate the results for four video sequences with missing data and corruption. From the table, we can see SSC with post-processing performs better relatively. LRSR performs better than LRR and SSC without post-processing.

The method been applied as partitioning dataset graph-based clustering class, and two steps to apply SC algorithms: subspace segmentation for constructing the affinity/similarity matrix and SC algorithms using the affinity matrix. However the implementation of SC not only to segmented and have graph-based class but also by embedding the data into **subspace** of ***eigenvectors*** of an ***affinity matrix***. Then to computing an affinity matrix **A** based on **S.A** , and must be made it of positive values and be symmetric. The optimization of statistical methods is usage of approaches such as the EM can leads to a local minimum, as statistical approaches normally use independent samples drawn from a mixture of probabilistic distributions to model data generation processes.

In this section, we evaluated the performance of our proposed algorithms on both synthetic data and the data will be exploited in next chapter for criminal data, then we will conclude the overall research and finding, then present our future work.

Chapter 5

Criminal Analysis with LRSR Clustering

This chapter will explain the tasks for the approach that been carried out to complete the project, and in fig. 5.2 we can see the Data-flow diagram and summary sketch for the tasks been carried out as the prototype integration for all phases as in fig. 5.1. As reviewed earlier in chapter 2, big data have one of the three main characters, **big quantities** refer to Volume, **high speed** of which data been generated called Velocity, and **different types** of data refer to it as Variety as in fig. 1.5. To evaluate the performance of any algorithms on big data such as crime data is best to understand the problem of the domain, examine and prepare the dataset, extract initial knowledge form the prepared/transformed dataset, then to apply an algorithm that will learn better, and last to discover and extract the knowledge.

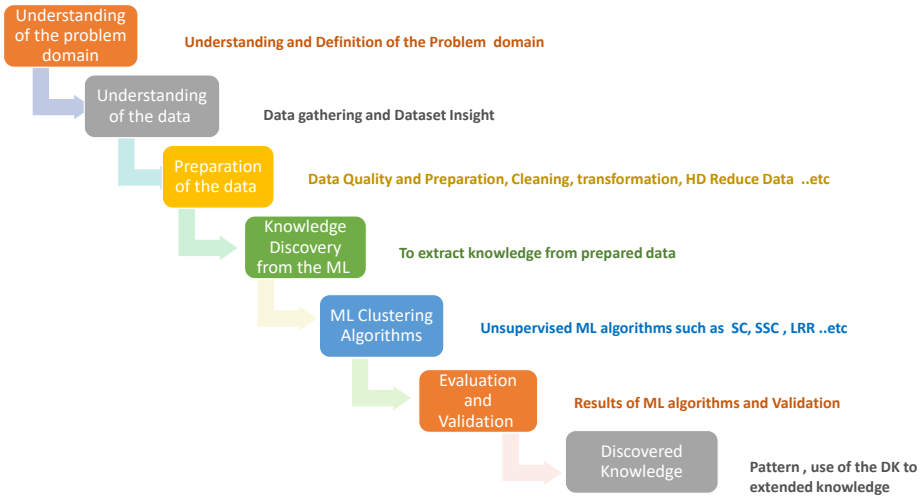


Figure 5.1 Knowledge Discovery Dataset and Machine learning **blueprint**

5.1 Knowledge Discovery Dataset and Machine learning Workflow.

Our Knowledge Discovery Dataset and machine learning module designed based on those **Five** elements that help in producing the following blueprint and core steps:

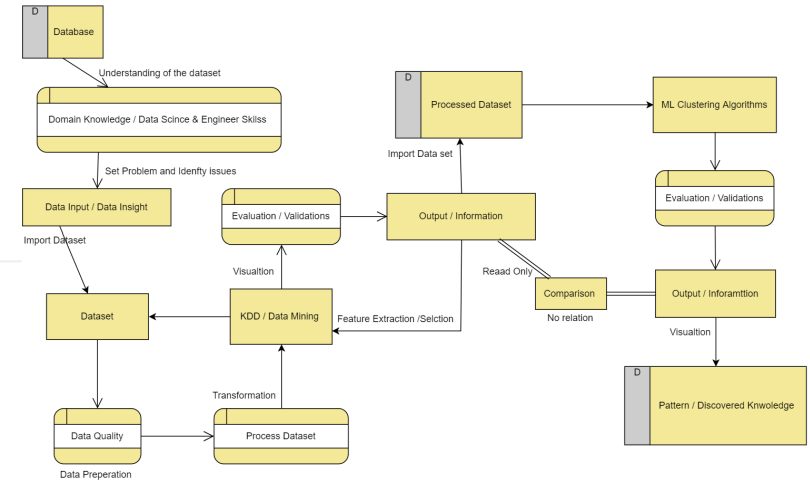


Figure 5.2 Knowledge Discovery Dataset and Machine learning Workflow for tasks been carrying out

1. **Exploratory Analysis:** First, "get to know" the data. This step was efficient, and critical.
2. **Data Preparation :** Include, cleaned, transformation our data to avoid many common pitfalls. that led to get the best of the algorithms.
3. **KDD and Feature Engineering:** helped our algorithms "focus" on what's important by creating new features.
4. **ML Clustering Algorithm applied** of the best most appropriate algorithms.
5. **Evaluation and validation** Result of our models, evaluate and validate task. This task will never be established if we have not complete the previous tasks.

In this section and based on what we have reviewed of techniques that exploited in chapter 2, we bring forth a practical approach that combined many approaches that used for both Criminal datasets analysis and big data. Theses fig. 5.1 techniques are the result of methods that exploited from both advanced ML algorithms and KDD "Data Engineers" that has

assisted in forming a strategy that bets dealt with crime data that has the characteristics of both big-data/high-dimensional dataset. We divided it into seven 7 tasks as in fig. 5.1. We have shaped and determined on these tasks form the outcome of the deep review and examine many approaches. for instance we have considered step 2, 3, and 4 from the KDD process as in fig. 2.10, as well as task 5, 6, 7 from Machine Learning algorithm methods and from data engineers. In addition, we have emphasized later on this section the benefit of using the above tasks before the advanced LRSR on Crime data. In this chapter, we will discuss how we have tackled the issue raised applying each task to such domain, as well in this section we will evaluate the result and how we have reached our the optimal module for big data. The following list is the strategies we accomplished that been divided into seven main tasks: 1) Understanding and Definition of the domain Problem. 2) Understanding of the Data, such as how was gathered , what insight we have , and how it visualize initially. 3) Data Quality , such as preparation and cleaning missing, duplicated values, Categorical data. 4) KDD , such as transformation and selection into target data to extract knowledge from the prepared data. 5) ML Clustering algorithms on crime data such as Unsupervised SC,SSC, LRR , LRSR algorithms. 6) Evaluate and Validation algorithms results. 7) Discovered Knowledge, Interpretation of the result such as patterns, and use of the Discovered Knowledge to extended knowledge. Next, each of the above tasks in fig. 5.1 applied, and discussed in following sections.

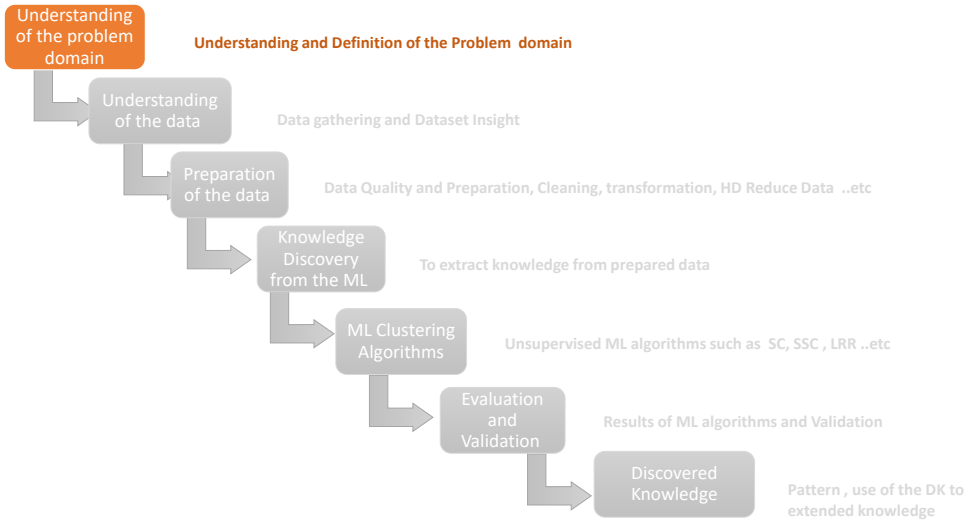


Figure 5.3 Understand the Problem domain

5.2 Understanding the Domain Problem

The objectives in this phase were to cover the domain problem in big and criminal data analysis. First, was the basic understanding of the dataset, and by identifying relation that is not clear on the dataset, rather what the dataset about or how it evolved. Thus, these fulfilled at each section for dataset expedients. However, there were critical questions been answered that justified our choice of which ML clustering methods been selected. It was a necessary task that equally important as applying, improving or evaluating algorithms. To understand the problem of any domain, these questions were answered: First, is to identify the information, shape, size, and the type of the original attributes of the whole dataset. Second is to specify the data type of each attribute. A third is to discover/deal with the missing/duplicated attributes.

5.3 Crime Datasets

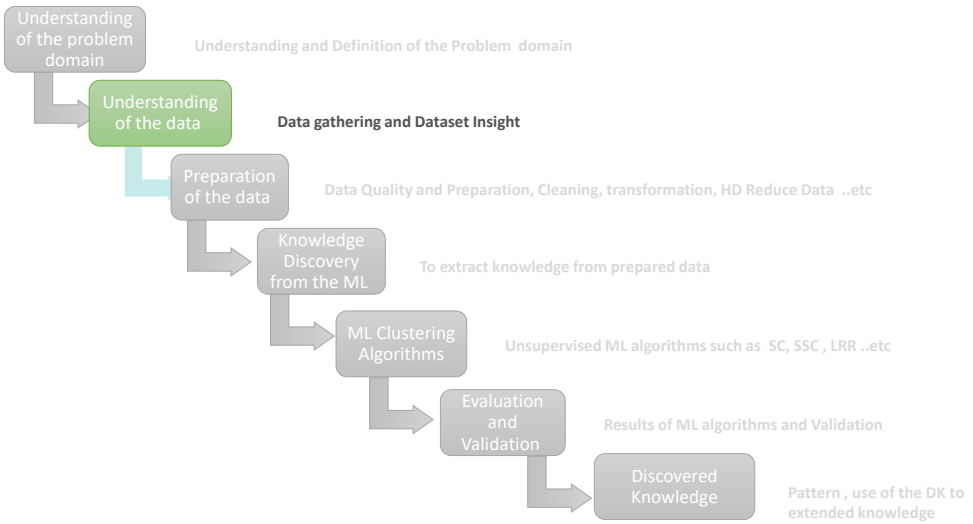


Figure 5.4 Understanding of the Dataset

Exploratory Analysis for Understanding the Dataset

We well first need to answer a set of questions about the dataset such as; How many observations do I have?, How many features?, What are the data types of my features? , Are they numeric?, Categorical? , and Do I have a target variable?

Data Gathering: experiment was on two dataset, the **first dataset** based on real crimes we will refer to it as **Local VALCRI Research Data**. The **second dataset** is **city of Chicago** crime data. This dataset reflects reported incidents of crime that occurred in the City of Chicago from 2001 to present. Data is extracted from the Chicago Police Department's CLEAR system. Running the experiments on synthetic datasets will not be best choice as real world data sets, as synthetic data sets may not well represent real world data, and both data sets has no label or class assigned to the instances. However

Data Insight

Best approach to get an close insight to what is the dataset about is to visualize the dataset before data preparation and quality tasks, and in this section we will visualize both dataset.

1. The VALCRI crime Dataset representative of a range of offence types that are appropriate to a typical offender for the major crime type, and the **Time Period** covered from April 2015 to March 2018. The VALCRI dataset Synthetic Data, and has 1.13M rows and 19 Columns. Each row is a Reported Crime. This dataset was based on real crimes that is of adequate size and complexity, and to subsequently develop to create and synthesise data that is good as the real data. This data set made available to the research community at Middlesex Unsisterly in London at other Organisation in European countries only.

2. The City of Chicago dataset initially can be described in table 5.2. The Chicago dataset has 6.79M rows and 22 Columns. Each row is a Reported Crime.

Table 5.1 **Both** Dataset Informations

	VALCRI dataset	**Chicago dataset**
Data type	float64(2), int64(3), object(14)	float64(4) , int64(6) , object(10)
Memory usage	1.4+ MB	1.5+ MB
Values	19	22
Rows	**1.13M**	6.79M

Table 5.2 Total record of all values presented in the **Both** dataset

VALCRI dataset			**Chicago dataset**		
Column	**∑ record**	**Type**	**Column**	**Total record**	**Type**
Crime Ref	10000	int64	ID	10000	int64
Crime Num	10000	object	Case Number	10000	object
Date First Committed	10000	object	Date	10000	object
Time First Committed	10000	object	Block	10000	object
Day First Committed	10000	object	IUCR	10000	object
Date Last Committed	10000	object	Primary Type	10000	object
Time Last Committed	8626	object	Description	10000	object
Day Last Committed	8626	object	Location Description	9981	object
Street	9976	object	Arrest	10000	bool
District	9074	object	Domestic	10000	bool
Town	9952	object	Beat	10000	int64
Post Code	7618	object	District	10000	int64
Offence	10000	object	Ward	10000	int64
MO Desc	1805	object	Community Area	10000	int64
Beat Num	10000	object	FBI Code	10000	object
HOMC Code	9419	float64	X Coordinate	9969	float64
HOOC Code	9419	float64	Y Coordinate	9969	float64
Northing	10000	int64	Year	10000	int64
Easting	10000	int64	Updated On	10000	object
			Latitude	9969	float64
			Longitude	9969	float64
			Location	9969	object

Table 5.3 VALCRI Example observations before ML

Record	**Values**
Crime Ref	81690360
Crime Num	99Y4/463197/19
Date First Committed	8-Mar-17 12:00:00 AM
Time First Committed	30-Oct-15 09:40:00 PM
Date Last Committed	18-Mar-17
Time Last Committed	30-Oct-15 09:55:00 PM
Day_Last_Committed	SAT
Day_First_Committed	SAT
Street	BUNTINGS LANE
District	LORDS ORCHARD
Town	CARSINGTON
Post_Code	B38 8AP
Offence	TAKE MOTOR/VEH W/O OWNER CONSENT
MO_Desc	BURGLARY ENTRY:REAR:WINDOW:CASEMENT:PLASTIC:SM
Beat_Num	E426
HOMC_Code	48.0
HOOC_Code	2.0
Northing	279100
Easting	403600

Table 5.4 Chicago Example observations before ML

Record	Values
ID	10000092
Case Number	HY189866
Block 0	47XX W OHIO ST
IUCR	041A
Primary Type	BATTERY
Description	AGGRAVATED: HANDGUN
Location	STREET
Beat	1111
Arrest	False
Domestic	False
Ward	28.0
Community Area	25.0
FBI Code	04B
X Coordinate	1144606.0
Y Coordinate	1903566.0
Year	2015
Updated	On
Latitude	41.891399
Longitude	-87.744385
Location	(41.891398861 -87.744384567)

Table 5.5 Chicago dataset attribute description

Column Name	Description	Type
Block	The partially redacted address where the incident occurred	Text
IUCR	Illinois Uniform Crime Reporting code	Text
Primary Type	The primary description of the IUCR code	Text
Description	The secondary description of the IUCR code	Text
Arrest	Indicates whether an arrest was made	Checkbox
Domestic	Indicates whether the incident was domestic-related	Checkbox
Beat	Indicates the beat where the incident occurred	Text
District	Indicates the police district where the incident occurred	Text
Ward	City Council district where the incident occurred	Number
Community Area	community area where the incident occurred	Text

Crime Dataset Example

This task will display example observations from both dataset. This will give us the "feel" for the values of each feature to check out if the values makes sense. The table 5.3 the examples of crime dataset values and entries of both data. **VALCRI** dataset has 19 entries , and there are very important entries, first, the type of **Offence** that have a total **168** different type Offences been found at VALCRI crime dataset; for example: " **TAKE MOTOR/VEH W/O OWNER CONSENT** " and it is one of the most important attribute, and there is an average frequent of **700** counts for each unique values. The second important attribute is the **MO_Desc** column that have a total **1407** different type crime Description that been found at VALCRI crime dataset; for example: " BURGLARY ENTRY : REAR : WINDOW : CASEMENT : PLASTIC : SM " which also one of the most important attribute in our dataset, and there is an average frequent of **37** counts for each unique values . The goal is to detect much more complex patterns where in real life there are numerous attributes for the crimes information available. In this table 5.3 values can be seen of each feature from VALCRI dataset before the process of KDD and ML been applied. In table 5.4 we can see the example of each feature from Chicago dataset before the process of KDD and ML been applied. For more description of Chicago city data, we including the attribute description in the following table 5.5.

5.4 Crime Data Preparation

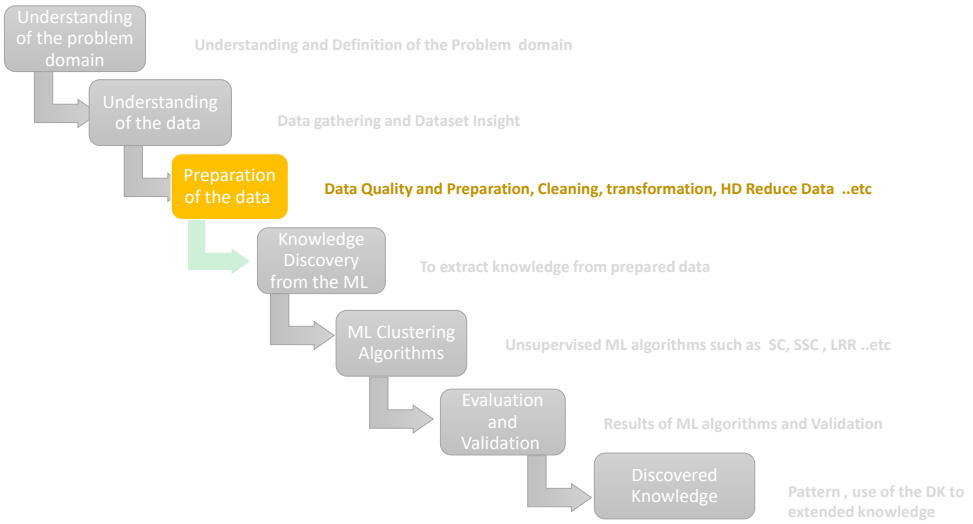


Figure 5.5 Data Preparation and Pre-processing

Data preparation is the third phase and tasks as in fig. 5.5 that we have included for our KDD and Criminal data analysis, known also as data pre-processing and data engineers. In this phase and after the data gathered and cleaned , the dataset need to be prepared in the format that ML can fit. We combined for this objective many tasks to solve the issues of Data Preparation and Pre-processing such as the Missing, Duplicated values, or outliers Values .

5.4.1 Crime Datasets Cleaning

Adopting this process it is very crucial to get the data ready for ML and to understand the domain, these process must be applied before any data reduction to learn about the data. The process that our domain that has undergone it has a significant positive impact on the final result for both dealing with crime big dataset and finding the best algorithm. Data preparation is a very important part of the machine learning process. Data Preparation was the most time-consuming aspect of this task. No matter how sophisticated the analysis technique that we have used, If we did not spend the time and effort to create good data for the analysis, we never got any good results. Therefore, in this section, we will illustrate how we make both

datasets in the right format for further analysis by solving both dataset quality issues as in the following sections

VALCRI and Chicago Missing Values

Missing data was existed in both data and it was a deceptively big issue in applied machine learning. First, we could not ignore missing values in our dataset, and we handle them carefully and individually for the very practical reason that most algorithms do not accept missing values. Most data science and engineers use the "Common sense" which is not reasonable in our domain. Data cleaning is time consuming and is the most important tasks before applying any ML algorithms. From our experience, the most worse commonly recommended ways of dealing with missing data is the dropping observations method which is harmful for the learning process. Dropping observations that have missing values is sub-optimal because when we drop observations, we then drop information, and in the real world, we often need to make predictions on new data even if some of the features are missing.

We needed to understand the data more by identifying the missing values, to do this we have first is to count the missing values using *insull* methods available to us in fig. 5.6 panda for DataFrame , where the counts that equal zero 0 is showing the feature that has no missing values . This method as in returned a boolean same-sied object indicating if values are null. we will then sum these values to find the column that have positive null values. Also we have visualize their distributions explained in section Figure 5.4.1 illustrated in fig. 5.7 .

Second, we have identified what type of the entry of each missing values, and then we found that our data contain both missing values of the numeric and categorical that have *NaN* and *empty* . We have split the tasks by type of missing values by their parent data-type. Missing values often been solved either by **Impute and Drop** methods , and as we mentioned that the use of dropping values or remove missing values it going to work in some cases when the proportion of missing values is relatively low $(< 10\%)$, but in our case it will make big impact and we then lose a large size of data. Likewise if we replace the values with **1** or **0** only then this will replace with $0(or-1)$. While this would help to run our models, but it can be extremely dangerous as sometimes these values can be misleading. Before display our result of our approach we need first to demonstrate and visualisation these missing values in next section followed by the method we employ and the result.

```
data.isnull().sum()
Crime_Ref                  0
Crime_Num                  0
Date_First_Committed       0
Time_First_Committed       0
Day_First_Committed        0
Date_Last_Committed        0
Time_Last_Committed     1374
Day_Last_Committed      1374
Street                    24
District                 926
Town                      48
Post_Code               2382
Offence                    0
MO_Desc                 8195
Beat_Num                   0
HOMC_Code                581
HOOC_Code                581
Northing                   0
Easting                    0
dtype: int64
```

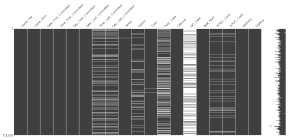
VALCRI

```
data.isnull().sum()
ID                         0
Case Number                0
Date                       0
Block                      0
IUCR                       0
Primary Type               0
Description                0
Location Description      19
Arrest                     0
Domestic                   0
Beat                       0
District                   0
Ward                       0
Community Area             0
FBI Code                   0
X Coordinate              31
Y Coordinate              31
Year                       0
Updated On                 0
Latitude                  31
Longitude                 31
Location                  31
dtype: int64
```

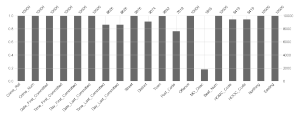
Chicago

Figure 5.6 *insull* methods identifying the missing values

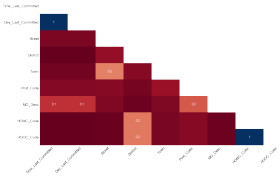
Plot the Numerical and Categorical missing values



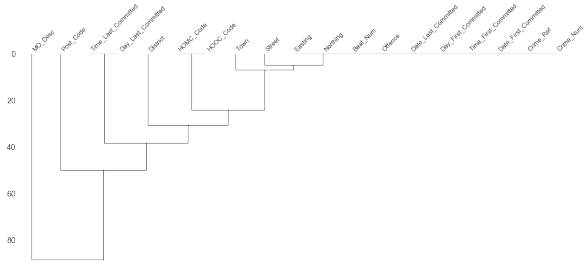
(a) Matrix



(b) Bar

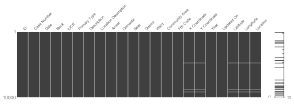


(c) HeatMap

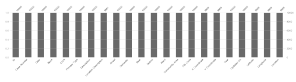


(d) Dendrogram

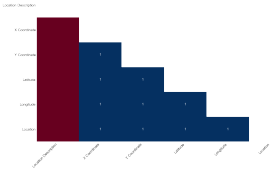
Figure 5.7 Missing values VALCRI Dataset



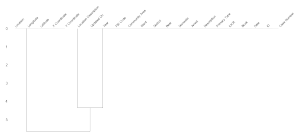
(a) Matrix



(b) Bar



(c) HeatMap



(d) Dendrogram

Figure 5.8 Missing values Chicago dataset

5.4.2 Handling Missing and Result

The graphs in fig. 5.7 and in fig. 5.8 shows the visualization of the present missing values of both dataset, and we presented using Matrix, Bar, HeatMap, and Dendrogram plot to have big picture of the data distribution. We have used the **HeatMap** as in fig. 5.7c also works great for picking out data completeness relationships between variable pairs, and the correlation heat-map measures nullity correlation on: how strongly the presence or absence of one variable affects the presence of another. As well we have used the the **Dendrogram** as in fig. 5.7d which allows us to more fully correlate variable completion, revealing trends

deeper than the pairwise ones visible in the correlation heat-map. We also add Bar Chart as in the is a fig. 5.7b for more visualization of nullity by column.

We could of use the IMPUTE methods but "missingness" is always informative in itself, and we have to tell our algorithm that we have value that is missing , so if we have built a model to impute our values, then these missing values is sub-optimal because the value was originally missing but if we filled it in, then it will leads to a loss in information, and no matter how advanced our imputation method was. Simply if Impute used, then we are not adding any real information, we just reinforcing the patterns already provided by other features.

Thus, How we handle missing Values? We inform our algorithm that a value was missing by: **1.)** We *Labelled* the **Missing categorical values** as $'Missing'$!. so we have essentially added a **new class** for the feature. This tells the algorithm that the value was missing. **2.) Missing numeric data** Flag the observation with an indicator variable of missingness in only targeted column, by for instant $df[1].fillna(0, inplace = True)$, which filled the original missing value with 0 to meet the technical requirement of no missing values. However, by using this technique of flagging and filling, we have essentially allowed the algorithm to estimate the optimal constant for missingness, instead of just filling it in with the mean, Std or most frequent values.

In the table bellow we can see as in fig. 5.9 that if the counts that equal zero 0 it is showing the feature that has no missing values. There are not too many missing values in **Chicago** dataset unalike **VALCRI** data that have many missing values. First we have find out number of missing values by which called *value_counts* as in fig. 5.9, for example in the city of Chicago dataset; the *Street* column have many missing Categorical values = **19** as in fig. 5.6, and we have found out the highest common values in that column is Categorical: "Location Description " which we use We *Labelled* the **Missing categorical values** as $'Missing'$!. In **VALCRI** dataset we found the *HOMC_Code* has total of = **581** missing Numerical values replaced by the 0 value

Finally, and after dealing with missing values in the **both** dataset we can see the result as in table 5.7. The result showing NO missing values exist. The graphs also in fig. 5.10 and fig. 5.11 shows the visualisation of the **VALCRI** and **Chicago** dataset after handled the missing values .

Table 5.6 *Labelled* **Missing categorical values** with *'Missing'*

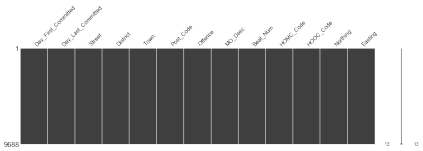
	Town_count	Town counts
0	LEAFORD	43
1	**Missing**	48
2	YEARSLEY	106
3	UPPER TURNBURY	133
4	INDORAM	146
5	YAFFORTH	153
6	CARNDON	167
7	TORMINGTON	187
8	TURNBURY	229
9	YUPPLETON	244
10	ELLIBOURNE	363
11	TILTON MARSH	409
12	YAULE	542
13	YARNFORTH	799
14	YORKTON	899
15	DEWMAPLE	1100
16	CARSINGTON	4432

```
data = data.apply(lambda x:x.fillna(x.value_counts().index[0]))
```

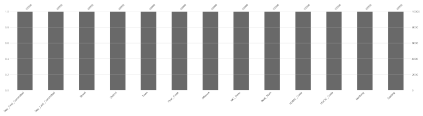
```
 Values after impute with most occuring   (9688, 13)

Day_First_Committed    0
Day_Last_Committed     0
Street                 0
District               0
Town                   0
Post_Code              0
Offence                0
MO_Desc                0
Beat_Num               0
HOMC_Code              0
HOOC_Code              0
Northing               0
Easting                0
dtype: int64
```

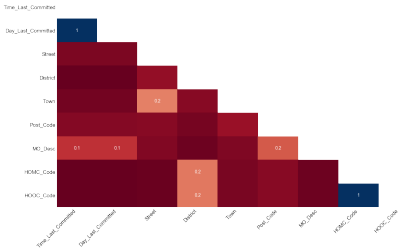
Figure 5.9 Result of Imputing all Missing values in VALCRI dataset



(a) A Matrix

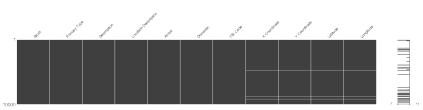


(b) Bar

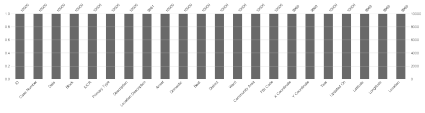


(c) A HeatMap

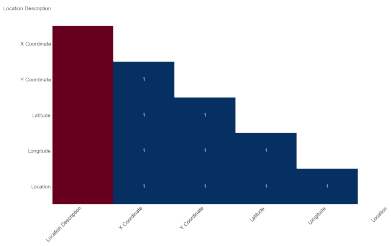
Figure 5.10 VALCRI dataset Visualization after dealing missing values



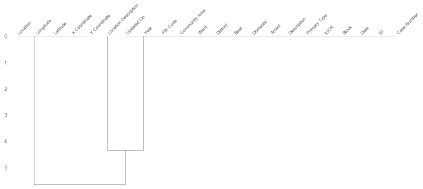
(a) A Matrix



(b) Bar



(c) A HeatMap



(d) A Dendrogram

Figure 5.11 Chicago dataset Visualization after removing missing values

After dealing with missing values in the **both** dataset as in table 5.7 .

Crime data Plot Numerical and Categorical Distributions

It was very helpful and revealing to plot the data to see the distributions of our both type of features. It was easy to deal with the numeric values but the categorical features cannot be visualized through histograms, Instead we use bar plots. We have mad notes about the result which has helped in choice most appropriate approach for furtherest ML transformation tasks. it dose cleared the invisible of seeing the large characterisation of the dataset, such as a potential outlier in one of our features that need to be more examine.

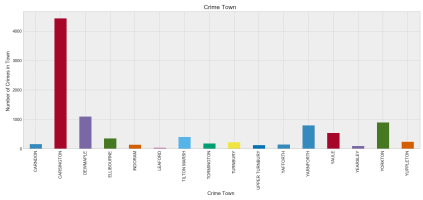
Table 5.7 Total record of both dataset after dealing with missing values

VALCRI dataset			**Chicago dataset**		
Column	**∑ record**	**Type**	**Column**	**Total record**	**Type**
Crime Ref	10000	int64	ID	10000	int64
Crime Num	10000	object	Case Number	10000	object
Date First Committed	10000	object	Date	10000	object
Time First Committed	10000	object	Block	10000	object
Day First Committed	10000	object	IUCR	10000	object
Date Last Committed	10000	object	Primary Type	10000	object
Time Last Committed	10000	object	Description	10000	object
Day Last Committed	10000	object	Location Description	10000	object
Street	10000	object	Arrest	10000	bool
District	10000	object	Domestic	10000	bool
Town	10000	object	Beat	10000	int64
Post Code	10000	object	District	10000	int64
Offence	10000	object	Ward	10000	int64
MO Desc	10000	object	Community Area	10000	int64
Beat Num	10000	object	FBI Code	10000	object
HOMC Code	10000	float64	X Coordinate	10000	float64
HOOC Code	10000	float64	Y Coordinate	10000	float64
Northing	10000	int64	Year	10000	int64
Easting	10000	int64	Updated On	10000	object
			Latitude	10000	float64
			Longitude	10000	float64
			Location	10000	object

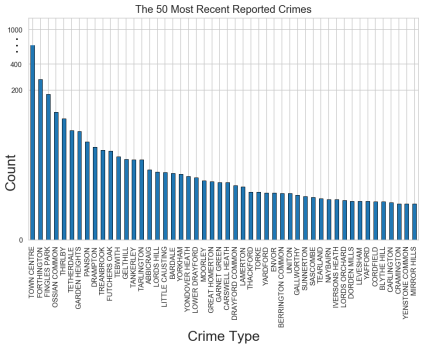
In particular, we looked out for sparse classes, which are classes that have a very small number of observations, and a "class" was result of finding the unique value for a categorical feature in both data. For example, the following bar plot will shows the distribution for all feature for all crimes in both dataset for each classes for each feature.

VALCRI Visualization and Distributions

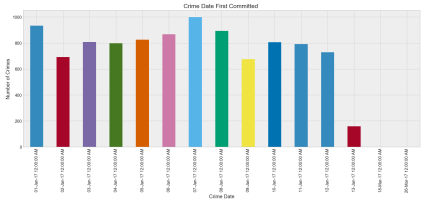
VALCRI dataset can be seen in fig. 5.12 that we can see the crime in District fig. 5.12b , town fig. 5.12a, at day fig. 5.12c , and in time.



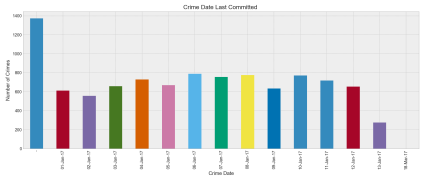
(a) Per Town



(b) Per District



(c) Per Date 1st Committed

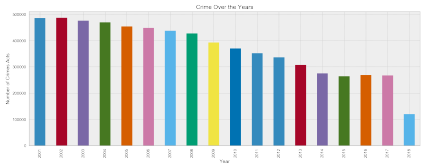


(d) Per Date last Committed

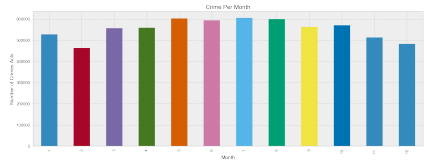
Figure 5.12 Crimes in VALCRI

Chicago City Visualization and Distributions

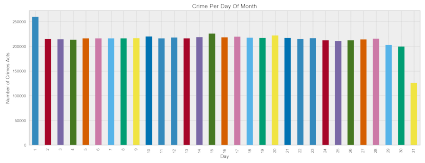
We can see the crime in **Chicago** City in fig. 5.13 over the years as in fig. 5.13a, Per Day Of Month as in fig. 5.13b, and Per Hour fig. 5.13c. We can as well there are different types of crimes as in fig. 5.14 which we separated them per month fig. 5.14a , also we have discovered what is the highest type of Crimes Committed in the city , as well as the relatives amounts of each type of crime as in fig. 5.14b. As well visualisation has helped us to see crimes and the arrest rates as in fig. 5.15 per community area fig. 5.15a, per ward fig. 5.15b, and per district of crimes committed in the city as in fig. 5.15c



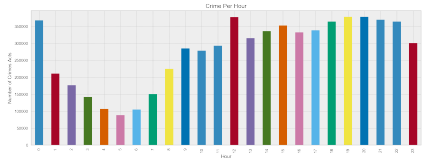
(a) crimes per year since 2011



(b) crimes Per Month all years



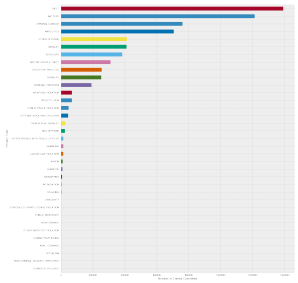
(c) crimes Per Day all years



(d) different crimes Per Hour all years

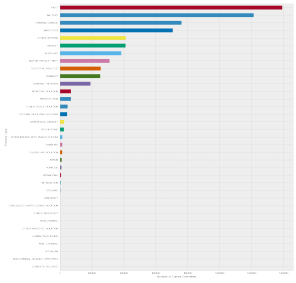
Figure 5.13 Chicago crime dataset visualization

(a) different of all Per month all years

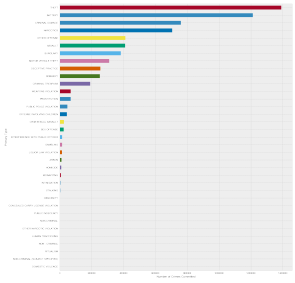


(b) different of crimes per month all years

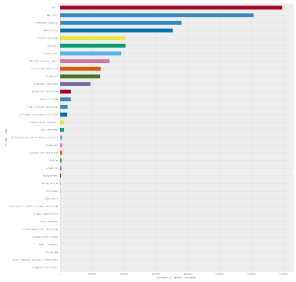
Figure 5.14 Different types of Chicago crime dataset



(a) Chicago city Arrest Rates Per District



(b) Chicago city Arrest Rates Per Ward



(c) Chicago city Arrest Rates Per Community Area

Figure 5.15 Arrest rates per communities of Chicago crime dataset

Dealing with Duplicate data

Dealing with duplicate data can occur when merging data from multiple sources. Here in table 5.12 we can see that in both dataset we have duplicate values in all records. Best approach but not in crime data is to remove the duplicated older record and keep the recent. However in our domain dealing with duplicated values is sensitive and if we have removed these values then the future result might be affected especially when analysing crime or finding pattern and relation in the dataset .

Values Dropping

We have dropped two particular columns from both data have unnecessary information such as Crime_ref, Crime_Num,MO_Desc, Northing, Easting, Time_First_Committed, and Time_Last_Committed as stated in table 5.3. These values dose not have a valuable contribute to the learning process based on the total count of each unique values and the class distribution explained in table 5.12. We have used function $data.drop()$ in $pandas.DataFrame$ library for data manipulation and analysis in python. As result of this task we have memory usage decrease from **1.2 MB** to **937.6+ KB** and data size of (**10000**, **12**). Nevertheless, at the end of this chapter a discussion of these result in details will be giving .

VALCRI dataset and Covariance

The problem with covariance to keeps the scale of the variables X and Y, which lead problems with Interpretation as in eq. (5.1) , because we can't compare variances over data sets with different scales such as time and longitude.

This makes interpretation difficult and comparing covariances to each other impossible. For example, $Cov(X,Y) = 5.2$ and $Cov(Z,Q) = 3.1$ tell us that these pairs are positively associated, but it is difficult to tell whether the relationship between X and Y is stronger than Z and Q without looking at the means and distributions of these variables. This is where **correlation** are useful by **standardizing** covariance by **Spearman's Correlation**.

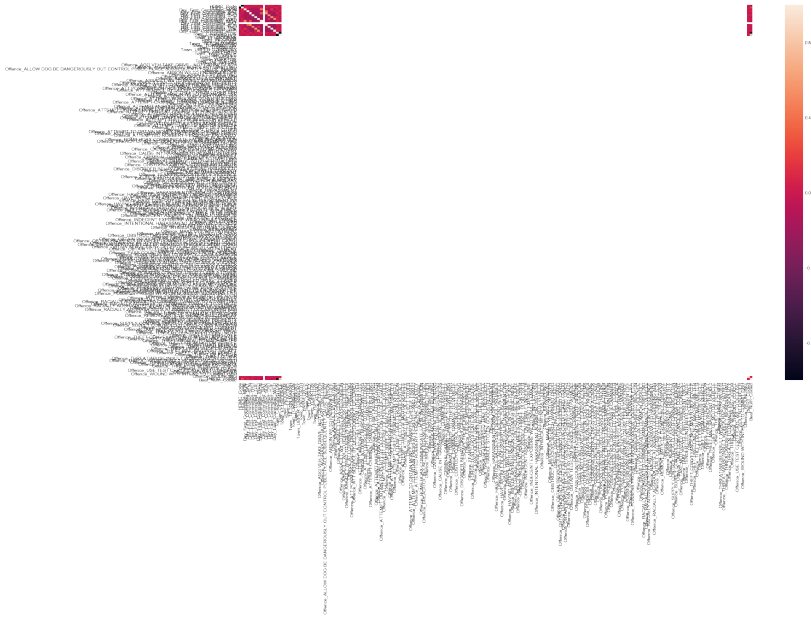
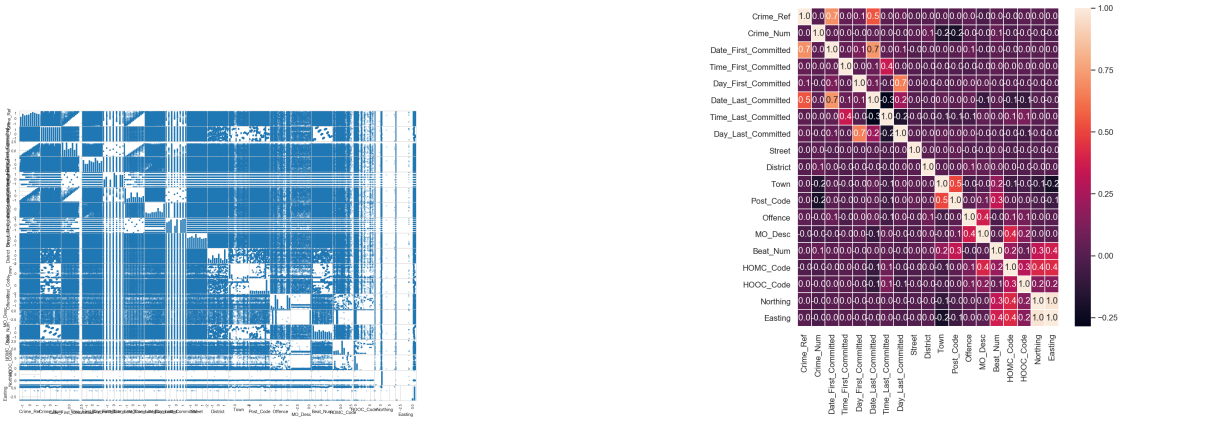


Figure 5.16 Correlation HeatMap for all dataset values in VALCRI

The number wont tell how strong that relationship is, but can be fixed by dividing the covariance which by the standard deviation to get the correlation coefficient. **Yet**, there were different result of correlation for the data before and after scaling . We can see in fig. 5.17a correlation among features before scaling less than once scaled as in **??**

$$cov_{x,y} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N-1} \tag{5.1}$$

The reason why we have chose to calculate the correlation coefficient as it has advantages over covariance for determining strengths of relationships. Covariance can take on practically



(a) Correlation After Scaled (b) Correlation Before Scaled

Figure 5.17 Arrest rates per communities of Chicago crime dataset

any number while a correlation is limited: -1 to $+1$. also, correlation is more useful for determining how strong the relationship is between the two variables ,and it does not have units, and finally , correlation is n't affected by changes in the center (i.e. mean) or scale of the variables. In next section we will explain the result of this method.

Knowledge Base Crime dataset Correlations distribution

Finally, Knowledge Base Crime dataset Correlations distribution an important step in our frame work that has allowed us to look at the relationships between numeric features and other numeric features before and after feature transformation.

What are Correlation in both dataset We have looked out for which features are strongly correlated with the other variable that we need to target, and is there any interesting or unexpected strong correlations between other features. Our aim was to gain more intuition about the dataset that helped us throughout the rest of the workflow.

So, What are the correlation in both dataset among attributes. why we looked for a mutual relationship or association between quantities in both dataset, as it assist in predicting one quantity from another and we will use it for many other modelling techniques in next chapter. We measured the association between random variables. There are several methods we used for calculating the correlation coefficient, each measuring different types of strength of association. Bellow, we show how to calculate correlation problem. First we were interested in explaining of correlation between all data values, and in fig. 5.16 we can observe the values by the Heat Map method.

Spearman's Correlation

We have investigate the correlation between the feature start by the exploit of the Spearman correlation which not only measure that calculates the strength of the relationship between the relative movements of the two variables , but also Spearman rank correlation coefficient can be defined as a special case of Pearson ρ applied to ranked or sorted variables. Unlike pearson, Spearman as is not restricted to linear relationships. Instead, it measures monotonic association and only strictly increasing or decreasing, but not mixed which between two variables and relies on the rank order of values. Where Pearson, as Pearson correlation measures the linear association between continuous variables. In other words, this coefficient quantifies the degree to which a relationship between two variables can be described by a line.

The formula we used for Spearman's as in eq. (5.2) coefficient looks very similar to that of Pearson as in eq. (5.3) , from the distinction of being computed on ranks instead of raw scores:

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \tag{5.2}$$

where d in eq. (5.2) = the pairwise distances of the ranks of the variables x_i and y_i . n = the number of samples, while the estimate for eq. (5.3) shown in .

$$\rho = \frac{\text{cov}(X,Y)}{\sigma_x \sigma_y} \tag{5.3}$$

the the estimate of eq. (5.3) :

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2(y_i-\bar{y})^2}} \tag{5.4}$$

The difference between Spearman and Pearson correlations is the Spearman's coefficient often show is **1** as in fig. 5.18a. where then Pearson correlation often weaker in this case as in fig. 5.18a, but it is still showing a very strong association due to the partial linearity of the relationship.

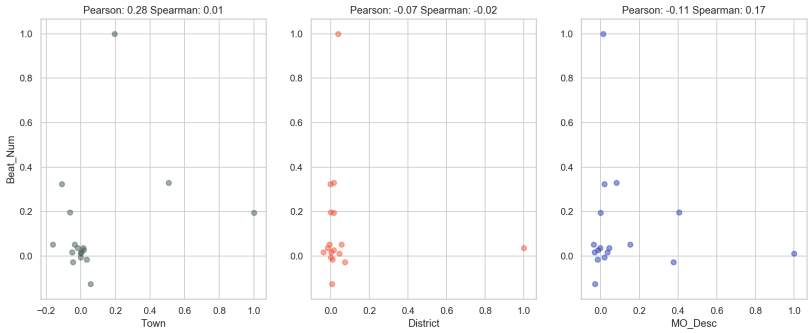


Figure 5.19 VALCRI non Scale Correlation

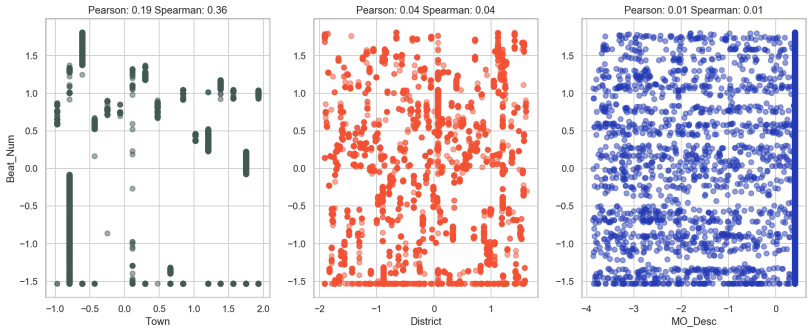


Figure 5.20 VALCRI Scale Correlation

	HOMC_Code	HOOC_Code	District_Coded	Beat_Num_Coded
HOMC_Code	1	0.089	0.017	-0.087
HOOC_Code	0.089	1	-0.025	-0.0033
District_Coded	0.017	-0.025	1	0.051
Beat_Num_Coded	-0.087	-0.0033	0.051	1

(a) Spearman correlations

	HOMC_Code	HOOC_Code	District_Coded	Beat_Num_Coded
HOMC_Code	1	0.076	0.0025	0.017
HOOC_Code	0.076	1	-0.0033	0.012
District_Coded	0.0025	-0.0033	1	0.048
Beat_Num_Coded	0.017	0.012	0.048	1

(b) Pearson correlations

Figure 5.18 Result difference between Spearman and Pearson correlations

VALCRI Correlation Vs non Scale Correlation result

We compute non Scale pairwise correlation of columns as in fig. 5.19, excluding NA/null values. Also we Compute correlation measures the linear association between continuous variables as in

Next we will introduce how Crime Data Transformation approach in High Correlation values

5.4.3 Crime Data Transformation

For both dataset was very helpful finding the *number* of **Unique** *Data Points* from **VALCRI** as in table 5.12 , then make these as dummies. The reason of why we have **Transformed**

String to numeric values and we have **converted categorical variable into dummy/indicator variables**, because most machine learning packages might transform categorical data to numeric automatically based on some default embedding method, many other machine learning packages don't support such inputs. In next sections we will explain our approach on dealing with these values

VALCRI categorical Transform

We have converted categorical variable into dummy/indicator variables. *Dummies* made by creating multiple dummy variables, that handling *categorical* features to be *numeric*. Then, we included a categorical values using dummy encoding FUNCTIONS. We have the following values: Day_First_Committed, Day_Last_Committed, Town, Date_First_Committed, and Date_Last_Committed .

VALCRI dummy encoding function We implemented by creating a new into separate new **df**, then **df1** = $data + dummies$ **df 2** = $df1 + dummies$ **df 3** = $df2 + dumies$. The data re-created in new **df** dataset by dropping the original 'columns name' column then join the new columns with the original dataframe as in fig. 5.22 , then the **df** has many columns. We create these dummy variables because Handling categorical features expects all features to be numeric. thus we can not include a categorical feature in our model, and we transform them to sensible numeric values by the use of dummy encoding. For example; leaving the encoding as 1 = Mon, 2 = Tue, 3 = Wed, and 4 = Thu ..etc. , have an issue of systematic order, and dummy encoding needed to have better and random learning and to avoid systematic learning. **Utility function** used to create dummy variable, then columns added for categories that only appear in our set then we added dummy columns to DataFrame as in fig. 5.21.

```
# An utility function to create dummy variable
def create_dummies( df, colname ):
    col_dummies = pd.get_dummies(df[colname], prefix=colname)
    col_dummies.drop(col_dummies.columns[0], axis=1, inplace=True)
    df = pd.concat([df, col_dummies], axis=1)
    df.drop( colname, axis = 1, inplace = True )
    return df
```

Figure 5.21 dummy encoding FUNCTION

	Street	District	Town	...	Day_Last_Committed_THU	Day_Last_Committed_TUE	Day_Last_Committed_WED
0	BUNTINGS LANE	LORDS ORCHARD	CARSINGTON	...	0	0	0
1	ADMINGTON	CRAMINGTON	CARSINGTON	...	0	0	1
2	COKERS LANE	NAYBAIRN	CARSINGTON	...	0	0	0
3	OCEAN CLOSE	ISTREAY BROOK	DEWMAPLE	...	0	0	1
4	HERBERT CRESCENT	CHARLMERE	YARNFORTH	...	0	0	0

5 rows × 23 columns

Figure 5.22 join the new columns

VALCRI Transform String to numeric values The following values: District and Beat Num, HOMC_Code, HOOC_Code, Offence, District, Beat_Num,Street, and Post Code have a **String** values as in table 5.9, and been transformed into Numerical values as seen in table 5.10 it will not prevent from been handled by the data mining and ML, because most of these techniques uses only numerical attribute. We replaced string by numeric which help us to see and to generates dataset's distribution that summarized data as in table 5.14. The different of dummy encoding used earlier that; replacing strings values with a encoding it will cause an order in the data, which then it will consider as a continuous numerical values that induce an order which simply should not exist in our dataset and avoided.

Table 5.8 String replace with numerical values in VALCRI

District	Beat Num
LORDS ORCHARD	E426
CRAMINGTON	E320n
NAYBAIRN	E536
ISTREAY BROOK	M541
CHARLMERE	H457

Table 5.9 String values

District	Beat Num
176	146
135	49
194	170
662	130
404	35

Table 5.10 Numerical values

The result of Transform String to numeric values and dummy encoding FUNCTION applied that we have created new columns that becomes an array = 654.4 KB and (10000, 62). However, a discussion of these result in details will be available at the end of this chapter.

VALCRI Transformation Categories to Numeric Values

In **VALCRI** dataset in table 5.2 we have these values as categories as *object* type : **District , Town , Street , Post Code , Offence , MO Desc, and Beat Num** types. Therefore we have transformed these categories values to numerical attributes as in table 5.10

5.4.4 Result of Dataset Transformation and Quality

This section demonstrate the result of the data quality and preparation tasks beforehand the KDD process and fitting the ML algorithms. Result presented in table 5.11 showing the result of the dimensions, size and data type after dataset gone under data quality, insight, preparation and values transformation tasks as part of the better understanding of the dataset. Many tasks has been implemented such as dealing with missing, duplicated, and categorical values. These result shows the improvement made on the datasets been ready for complex tasks such as transformation , learning , reduction and Clustering. The result of these tasks has helped in finding the best data representations of both crime dataset that led to fit the data ML algorithms.

Name of New data	Columns	dtypes	memory usage
data	19	float64(2), int64(3), object(14)	1.4 MB
ScaleDum	186	float64(185), int32(1)	14.2 MB
ScaleNoDum	20	float64(19), int32(1)	1.5 MB
DumNotScaleNum	225	int64(225)	17.2 MB
DumNotScaleClean	186	float64(2), int32(1), int64(183)	14.2 MB
NoDumNoScaleNum	20	int32(1), int64(19)	1.5 MB
NoDumNoScaleNotNum	19	float64(2), int64(3), object(14)	1.4 MB

Table 5.11 Data Quality and Transformation task Result

In next section we will move further to advanced data pre-processing for transformation and data selection such as normalization, standardization, data selection, and data reduction.

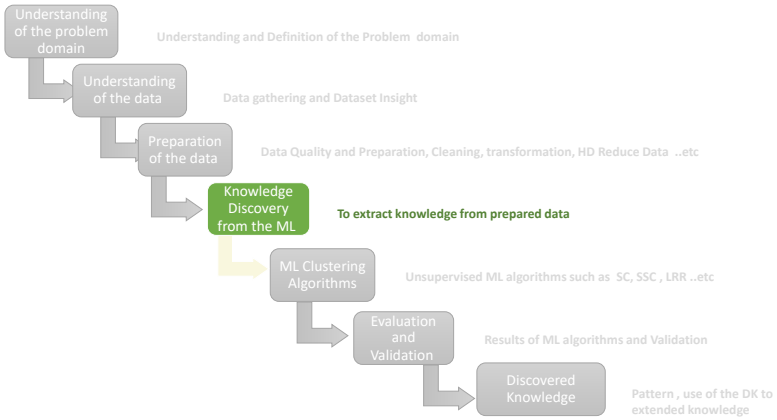


Figure 5.23 Knowledge Discovery

5.5 Knowledge Discovery Dataset

This section covered the knowledge discovered from the Pre-pared dataset to be preprocessing values into processed data fit to ML algorithmics experiments. This section presented how features selected form transformed and prepared dataset, as well as presented the identified related features by the use of both statistical and correlation distribution that identifies the high correlated attributes, then we will discussion how we extracted features and knowledge form the processing dataset by the Normalization, and standardization approaches. In next section we will start by viewing the big picture and the best way is to begin with virtualisation of both dataset to view the respective values for each task.

5.5.1 Crime Dataset Class Distribution

In this task we are demonstrate the number of instances (rows) that belong to each class. We can view this as an absolute count as table 5.12. For example, we can see from VALCRI dataset; that each class has almost different number of instances (max 10000 and min 7 and avarage of of the dataset). However, this dose not tell us anything about the data ,but it can assist to have an insight how the data was and in feature analysis. The following section will demonstrate how we learn more about the datasets and how has helped us in finding out what are the **distribution** of the numerical feature of each values across the samples of both dataset as in early insights. As well the insight and it's visualisations will be shown and after the results.

Dataset and shape description In both dataset we determine the over-fitting values by the use shape of a dataset's distribution as it help to over come the inability to analysis the dataset, so for description and shape distribution we found the **VALCRI** dataset it has string values as in table 5.10 that unable us to generates **descriptive statistics** that summarize the central tendency, dispersion and shape of a dataset's distribution solved in the next section.

Statistical Summary Both dataset prior to transformation This task is to observe at a summary of each attribute, we have a very large standard deviation in both data , but isn't necessarily a negative or harmful to the ML module as it has the **same units** as the **original** data ; as it often reflects a large amount of variation in the group and calculated and the fact that it measures a distance from the mean such as the **Crime_Ref** value in VALCRI dataset seen in the table 5.13 . Also this step and observation have confirmed that our data have an outliers which we going to cover in separate section, and once the Std. it's almost as large as the average then we haev an outlier which it was the case in VALCRI data and alomots for all values . For instant: the **Northing** values of the VALCRI dataset was 6.339500 , and the standard deviation of the entire record turns out to be 10.740371.

Subset of each values in both dataset If we look to find features or subset in high dimension then we need first to look at the good subset, and the best is the one least number of dimensions that most contribute to learning accuracy, thus we have observed and looked at the number of unique Data Points of each values at the VALCRI dataset we can see that there is some values that have lower data points to be consider in future learning accuracy as we see in table 5.12. Chicago city on the other hand, have also number of unique Data Points of each values as subset, we can see that there is some values that have lower data points to be consider in future learning accuracy as we see in table 5.16. We can see in fig. 5.25 the pie chart and how the percentage of each total number of subset of each values represented in VALCRI, which has helped on seeing the board picture of the distribution.

Table 5.12 Class Distribution of Both dataset Data Points numbers of each values

VALCRI dataset		**Chicago dataset**	
Column	**∑ unique Points**	**Column**	**∑ unique Points**
Crime_Ref	10000	ID	10000
Town	16	Date	5060
Street	4365	Block	6645
Post Code	5564	IUCR	218
Offence	168	Description	208
MO_Desc	1407	Domestic	2
Beat_Num	682	Beat	277
Day_First_Committed	7	District	22
Day_Last_Committed	7	Ward	50
District	300	Year	8
Crime_Num	10000	Latitude and Longitude	8720
Date_First_Committed	15	Location	8720
Time_First_Committed	684		
Date_Last_Committed	15		
Time_Last_Committed	780		
HOMC_Code	60		
HOOC_Code	51		
Northing	323		
Easting	456		

Table 5.13 Statistical Summary of VALCRI Dataset

	Crime_Ref	**HOMC_Code**	**HOOC_Code**	**Northing**	**Easting**
count	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000
mean	1.235783e+08	247.776800	6.339500	266288.150000	376033.670000
std	6.425597e+05	8787.140178	10.740371	77747.511254	108600.244607
min	8.169036e+07	1.000000	0.000000	0.000000	0.000000
25%	1.231698e+08	34.000000	2.000000	281400.000000	396300.000000
50%	1.236152e+08	48.000000	3.000000	286600.000000	404900.000000
75%	1.239961e+08	58.000000	10.000000	291900.000000	411400.000000
max	1.244137e+08	399898.000000	105.000000	880200.000000	438900.000000

H

Figure 5.24 Both dataset Data Points numbers of each values

Table 5.14 Statistical Summary of Chicago Dataset

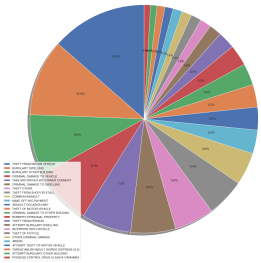
	Beat	**District**	**Ward**	**Community Area**	**Year**
count	6.71E+06	6.71E+06	6.10E+06	6.09E+06	6.71E+06
mean	1.19E+03	1.13E+01	2.27E+01	3.76E+01	2.01E+03
std	7.03E+02	6.95E+00	1.38E+01	2.15E+01	4.97E+00
min	1.11E+02	1.00E+00	1.00E+00	0.00E+00	2.00E+03
25%	6.22E+02	6.00E+00	1.00E+01	2.30E+01	2.00E+03
50%	1.11E+03	1.00E+01	2.20E+01	3.20E+01	2.01E+03
75%	1.73E+03	1.70E+01	3.40E+01	5.80E+01	2.01E+03
max	2.54E+03	3.10E+01	5.00E+01	7.70E+01	2.02E+03

Table 5.15 number of subset of each values in VLACRI

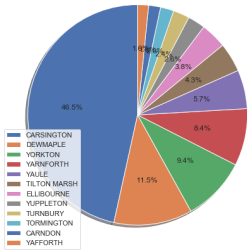
	A) Offence crime	**counts**		**B) Town crime**	**counts**
0	THEFT MOT. VEHIC.	1216	0	CARSINGTON	4432
1	BURGLARY DWELLING	971	1	DEWMAPLE	1100
2	BURGLARY BUILDING	768	2	YORKTON	899
3	DAMAGE TO VEHICLE	726	3	YARNFORTH	799
4	TAK.VEH OWN. CONS.	631	4	YAULE	542
	C) District crime	**counts**		**D) Street crime**	**counts**
0	TOWN CENTRE	967	0	HIGH HALDEN LANE	226
1	FORTHINGTON	286	1	COLWINSTONE ROAD	75
2	FINGLES PARK	195	2	IVESON GREEN	64
3	OSSIAN COMMON	171	3	CLARKE BROW	59
4	THIRLBY	162	4	YARROW CRESCENT	57
	E) Post_Code crime	**counts**		**F) Day_First_Committed**	**counts**
0	DY5 1SY	27	0	SUN	1831
1	B76 8XL	17	1	TUE	1618
2	B26 3QJ	17	2	WED	1593
3	WS1	15	3	THU	1557
4	B10 0HH	15	4	MON	1372
	G) Day_Last_Committed	**counts**		**H) Beat_Num**	**counts**
0	WED	1451	0	-	584
1	TUE	1435	1	F324	106
2	SUN	1390	2	M324	89
3	THU	1328	3	F306	89
4	MON	1195	4	H316	83
	I) Date_Last_Committed	**counts**		**MO BURGLARY Desc_count**	**counts**
0	-	1374	0	ENTRY:WOOD:FORCED...	37
1	06-Jan-17	791	1	ENTRY:FRONT:DOOR...	14
2	08-Jan-17	777	2	ENTRY:FRONT:FORCE...	12
3	10-Jan-17	774	3	ENTRY:FRONT:CASEMENT...	11
4	07-Jan-17	757	4	ENTRY:FRONT:DOOR:...	10

Table 5.16 number of subset of each values in Chicago

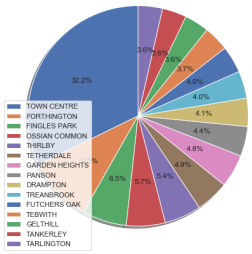
	A) Year crime	**counts**		**B) Ward crime**	**counts**
0	2002	486752	0	28.0	278114
1	2001	485745	1	42.0	243700
2	2003	475934	2	24.0	239304
3	2004	469380	3	2.0	236533
4	2005	453703	4	27.0	220424
	C) District crime	**counts**		**F) Beat crime**	**counts**
0	8.0	456941	0	423	52343
1	11.0	428814	1	421	51401
2	7.0	396775	2	624	46471
3	25.0	387891	3	1533	45395
4	6.0	386346	4	511	45071



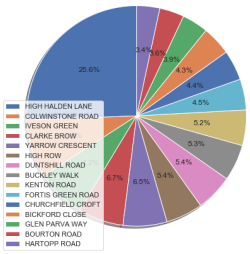
(a) Offence Unique values



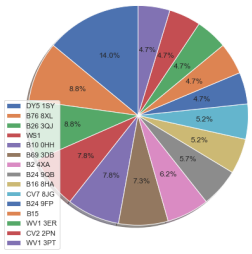
(b) Town Unique values



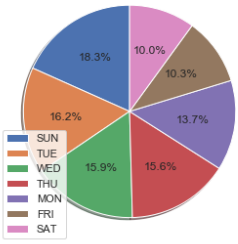
(c) District Unique values



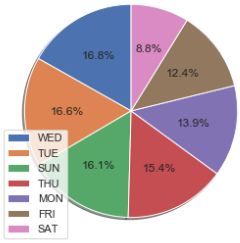
(d) Street Unique values



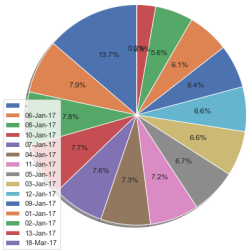
(e) Post Code Unique values



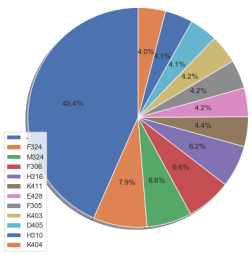
(f) Day 1st Com Unique values



(g) Day Lst Com Unique values

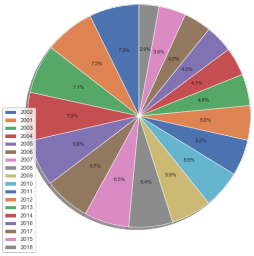


(h) Data Lst Com Unique values

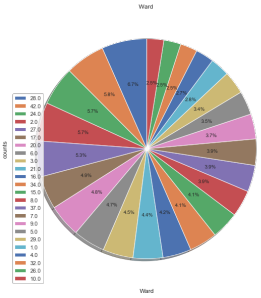


(i) Baet Num Unique values

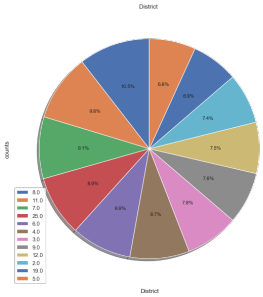
Figure 5.25 Total of number of each Unique values VALCRI dataset



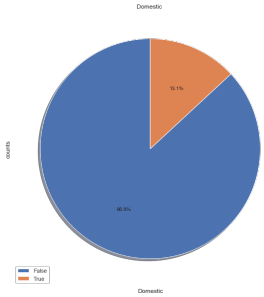
(a) Year Unique values



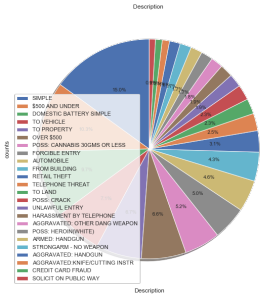
(b) Ward Unique values



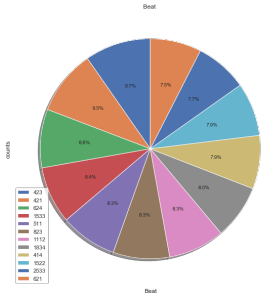
(c) District Unique values



(d) Domestic Unique values



(e) Description Unique values



(f) Beat Unique values

Figure 5.26 Total of number of each Unique values Chicago dataset

VALCRI and Chicago City Outliers

Outlier in our data have the values that are considerably different than the rest of the other data samples in a data set. Here an example of how we detected Outliers through the use of summary plots of the data before applying our approach, for instant and In PARTICULARLY form VALCRI dataset as in fig. 5.27 we found the **Offence** value as in fig. 5.27a and **District** as fig. 5.27b have an outliers among the other values where the **Town centre** is the highest record of the **Offence** value and **Theft from motor vehicle** are the highest of the **District** value. As well we have outlier in **Day Last Committed** value where the **WED** appeared **2726** times becoming the highest in the record as in fig. 5.27c

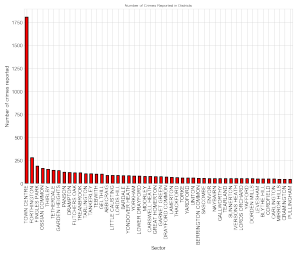
As we seen in above graph 5.27 that we have an Outliers in Offence, District, Town Day Last/First Committed, Twon, and Street Values. we acknowledge that most of engineers they tend to remove outliers regardless their values or meaning in the domain, but we confirm that not all outliers are undesirable. In **VALCRI** dataset Outliers can significantly important and are exactly what we're looking for, for example in fig. 5.27e we have an outlier represent the most Towns associated with most crimes, so in this can we must keep the town values with it outliers, So when detected outliers, we didn't move them out. Instead, we examined them more closely and we will show how we dealt with these values to make these values of these attribute available to analysis in next section.

Moreover, we found an attributes in **VALCRI** dataset have undesirable outliers and it can increases the error variance and reduces the power of statistical tests as in fig. 5.27g where it has numeric entries and have high unique numbers **300** by **750** counted for each of these unique number . Therefore In this cases these outliers are not the focus of our analysis, and we removed these outlier from our data set.

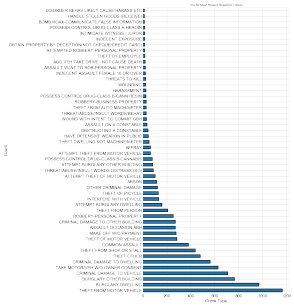
In next section , we will display tasks that involves deciding on which features to use , removed, added or combined to extracted important, non-redundant features from our data.

VALCRI and Chicago Over-fitting

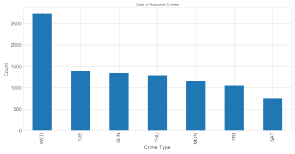
We aimed to avoid over-fitting in the final output of the pre-proceed tasks, which is one of the most harmful pitfalls in machine learning, because it will lead to an over-fit model that has "memorized" the noise in the dataset, instead of learning the true underlying patterns. For most applications, the stakes won't be quite that high, but in big data and more precisely crime data we often find an over-fitting that still the largest issue must avoid. We'll use few



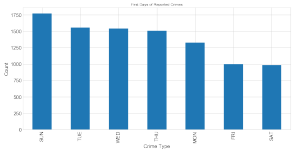
(a) Number of Crimes Reported in District Values



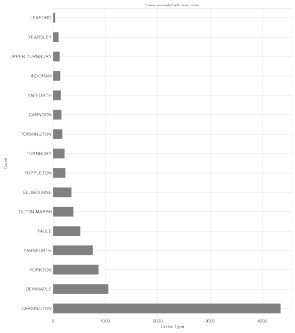
(b) The 50 Most Recent Reported Crimes Offence



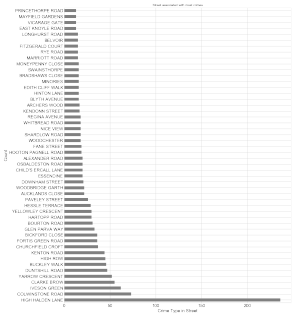
(c) Days of Reported Crimes in Day Last Committed Values



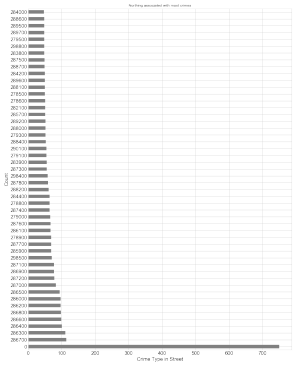
(d) Day First Committed Values



(e) Towns associated with most crimes in Town Values



(f) Street associated with most crimes in Street Values



(g) Northing Values

Figure 5.27 VALCRI outliers

strategies for preventing over-fitting first, by choosing the right algorithms and then tuning them correctly.

In this section we will cover how we selected data that are not over fitting and prepare our dataset into target data, and we have found the **Street, Post Code, and MO Desc** values from **VALCRI** dataset have factors which have over 1000 unique labels as in table 5.12, and it cause over-fitting for the model making to not be able to learn and understand the data well , and if we keep these values it will over-fit.

5.6 Crime Dataset Feature Engineering

Feature extraction and dimension reduction can be combined in one step using principal component analysis (PCA), linear discriminant analysis (LDA), or Recursive Feature Elimination (RFE). We aimed to apply Feature extraction to select features as a pre-processing step followed by clustering on feature vectors in reduced-dimension space.

The reason extract features because we have high dimensional dataset, and when the dimensionality increases the computational cost also increases, usually exponentially, besides we have in our dataset many attributes not yet suitable for feature selection or reduction task. To overcome this problem it is necessary to find a way to reduce the number of features in consideration. Two techniques we used (1) Feature subset selection. (2)Feature extraction.

This section will demonstrate how we first have standardize, normalize the dataset before presenting the reduced method we have applied followed by the reduced features result. This section will also explain the different results obtained from normalised and scaled and Feature extraction and dimension reduction .

5.6.1 Crime data Feature Transformation

At this point, both dataset been through very important tasks such as data quality, and preparation tasks that produced data clean, non-missing , non-duplicated , and non-categorical values that allowed us to improve the dataset even further. In this **section** we are going to illustrator how we have transformed the existing values into usable and useful features for ML tasks. The transformation of the data going to help in improving the accuracy of the algorithm later.

VALCRI Standardising and Scaling

Even we have dealt with missing ,duplicated, strings, categorical values, yet still some issue need to be resolved such as having attributes as in table 5.17 that not yet suitable for ML tasks such feature selection, Extraction or reduction. These values are not in one unite scale and would have different result even if we wanted to measure the relationship between them or operate a simple Statistical calculation such as finding the Distribution of the dataset to finds **Std.** or **Mean** as in table 5.25.

Approach We dealt with the issue by standardising and scaling dataset by Standardize features by removing the mean and scaling to unit variance. This approach standardization is so-called **Min-Max** scaling as in eq. (5.6). It is one of the Two common feature transformations. This step is very important when dealing with parameters of different units and scales, to reduce noise and variability to make the data easier to analyse. The standard score of a sample 'x' is calculated as in eq. (5.5) :

$$z = (x - u)/s \tag{5.5}$$

where 'u' is the mean of the training samples or zero , and 's' is the standard deviation of the training samples or one. Centring and scaling happened independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using the '$transform$' method used in Python as in.

Bottom-up approaches:

We have code the following equations for standardization and scaling. However, in this approach, we scaled the data to a fixed range. A Min-Max scaling is done via the following equation:

$$Xnorm = \frac{X - Xmin}{Xmax - Xmin} \tag{5.6}$$

$$\bar{X} = \frac{\sum x}{n} \tag{5.7}$$

Table 5.17 VALCRI Data **Non-Scaled** scaling Feature

Time_LasT	Street	District	Town	Post_Code	Offence	MO_Desc	Beat_Num
586	663	146	1	1887	137	1407	176
779	27	49	1	1491	50	1407	135
732	977	171	1	228	50	1407	194
779	2921	130	2	4455	45	1407	662
749	1933	35	12	4633	48	1407	404

$$Var(X) = \frac{\sum x^2}{n} - \bar{x}^2 \tag{5.8}$$

$$Var(Y) = \frac{\sum x^2}{n} - \bar{x}^2 \tag{5.9}$$

This approach simply Standardization refers to shifting the distribution of each attribute to have a mean of zero and a standard deviation of one (unit variance). It allowed us to achieved Standardize features that by removing the mean as in eq. (5.7) and scaling it to unit variance as in eq. (5.8) and calculate covariance as in eq. (5.9). indeed as we seen first by finding the (1.) Mean of each values for example (Offences after dummies against the other values), then use that mean to calculate the (2.) variance of selected values then (3.) covariance between each of them , and we can see the result on VALCRI data. In this table table 5.17 we can see the dataset before scaling , and in this table 5.18 we can see after the scaling.

Table 5.18 VALCRI Data **AFTER** scaling Feature

Time_Last	Street	District	Town	Post_Code	Offence	MO_Desc	Beat_Num
0.541707	-1.191202	-0.189722	-0.794902	-0.817479	0.962637	0.403912	-0.668432
1.357332	-1.704752	-1.332349	-0.794902	-1.068313	-0.755064	0.403912	-0.869735
1.158708	-0.937657	0.104770	-0.794902	-1.868323	-0.755064	0.403912	-0.580055
1.357332	0.632063	-0.378197	-0.613202	0.809144	-0.853782	0.403912	1.717740
1.230551	-0.165717	-1.497265	1.203800	0.921893	-0.794551	0.403912	0.451007
1.289715	-0.815729	1.577227	-0.613202	0.287839	-0.755064	0.403912	1.663732
-1.896714	-0.538767	0.045871	1.203800	0.999170	0.172890	0.403912	0.470646
1.357332	1.621212	-1.520824	-0.794902	0.809144	1.258792	0.403912	-0.555506
-1.926296	-0.281185	1.117821	-0.613202	0.133918	-0.893269	-0.949981	1.590084
1.357332	0.326839	1.200278	0.476999	0.745802	-0.735320	0.403912	0.691588
1.357332	0.531129	0.964685	-0.794902	-1.450899	0.962637	0.403912	-1.164324
1.357332	-0.052671	-0.236841	-0.794902	0.809144	-1.722504	0.403912	1.315135

```
%\begin{lstlisting}
import numpy as np
import pandas as pd
import sklearn
from sklearn.preprocessing import StandardScaler, MinMaxScaler

# Standardize time series data
from pandas import Series
from math import sqrt

# load the dataset
series = Series.from_csv("IntNoDum.csv",error_bad_lines=False,nrows =
    10000)

# prepare data for standardization
values = series.values
values = values.reshape((len(IntNoDum), 1))

# train the standardization
scaler = StandardScaler()
scaler = scaler.fit(IntNoDum)
print('Mean: %f, StandardDeviation: %f' % (scaler.mean_, sqrt(scaler.
    var_)))

# standardization the dataset and print the first 5 rows
normalized = scaler.transform(IntNoDum)
```

```
for i in range(5):
print(normalized[i])

# inverse transform
inversed = scaler.inverse_transform(normalized)
for i in range(5):
print(inversed[i])

Return :

Mean: 4999.500000, StandardDeviation: 2886.751332
[1.55105151]
[1.32034234]
[1.66017071]
[-0.48272256]
[-0.96631115]
[9477.]
[8811.]
[9792.]
[3606.]
[2210.]
```

Listing 5.1 Valcri Dataset Standardisation

5.6.2 Crime Dataset Feature Extraction

Data Reduction of the transformed data

In previous section we have solved the values that are not in one unite scale , still we have high dimensional data need to be reduce in order to discover pattern or relation among features. Next section will illustrate our approach in crime data domain.

Feature Extraction With PCA Choosing appropriate features to building better ML module started earlier in this chapter, all these tasks help our knowledge of domain increase which has been the key role in understanding of the application in deciding features to add, drop or modify. We reduced the FEATURE VECTOR by building the derived values to came up with the smallest set of features that have the best captures of the characteristics of our domain and tested through PCA explain in next paragraph.

	Crime_Ref	Post_Code	Offence	...	MO_Desc	Beat_Num	**kmeans**
0	81690360	1887	137	..	1407	176	**2**
1	122679527	1491	50	..	1407	135	**0**
2	122679664	228	50	...	1407	194	**0**
3	122679801	4455	45	...	1407	662	**0**
4	122680623	4633	48	...	1407	404	**0**

Table 5.19 VALCRI Before reduction PCA through KDD

VALCRI Data Reduction based PCA Data Reduction in many ways and we chose feature selection known also as Attribute Subset Selection , we applied PCA Projection to the original data which has 19 columns as in table 5.2. The original data projected into new 2 dimensions, we earlier have specified in advance the number of principal components we wanted to use. Then we can just call the fit() method with our data frame and check the results. and there usually no a particular meaning assigned to each principal component after dimensionality reduction. The new components is the two main dimensions of variation. We have exploited (PCA) as feature extraction method. We fond the eigenvectors of a covariance matrix with the highest eigenvalues as in table 5.22 and we used those to project the data into a new subspace of equal or less dimensions .

We have converted the matrix of n features into a less than n features as new dataset. That have reduced the number of features by constructing a new, smaller number variables which capture a significant portion of the information found in the original features. However, the goal of this section not to explain the concept of PCA, but rather to demonstrate the result of PCA in scaled and non-scaled crime dataset.

Principal Component Analysis applied by the following Scatter matrix:

$$S = \sum_{k=1}^{n} (\boldsymbol{x}_k - \boldsymbol{m})\ (\boldsymbol{x}_k - \boldsymbol{m})^T \tag{5.10}$$

where

$$\boldsymbol{m} = \frac{1}{n} \sum_{k=1}^{n} \boldsymbol{x}_k \text{ (mean vector)} \tag{5.11}$$

Table as in table 5.22 show the original **data** as in table 5.19 with 19 features then reduced by PCA explained by eq. (5.10) , and fit the *n_componenet* = 50 . Graph in fig. 5.28 shows the data reduced before dealing with missing values , features creation, transformation such as scaling or standardising .This graph shows the number of components as X label and

cumulative explained variance ratio as in Y label, which explain the parameter returns a vector of the variance explained by each dimension. Thus PCA explained variance ratio = $[i]$ gives the variance explained solely by the $i + 1$ st dimension. Also explained variance ratio can be derived from singular values. However the graph in fig. 5.29 we can see the VALCRI data reduction after cleaning dataset , dealing with missing values , categorical data type , and scaling to one unite variate.

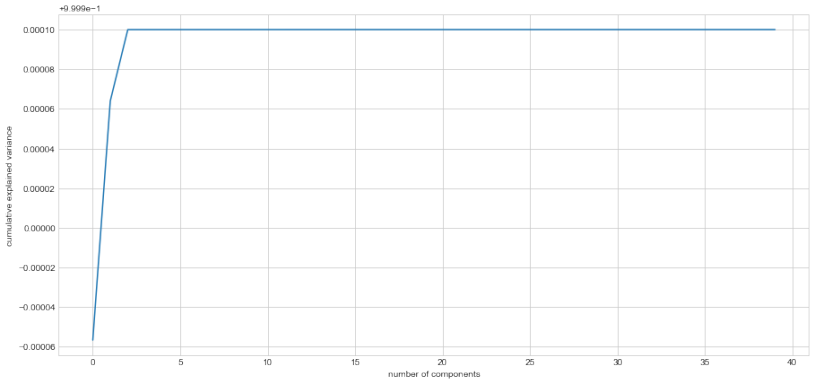


Figure 5.28 VALCRI data reduction Before KDD

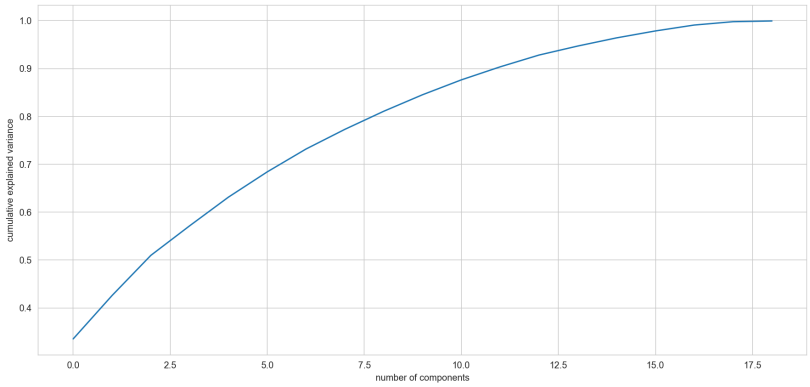


Figure 5.29 VALCRI data reduction After KDD

This table 5.20 represent the The new components as two main dimensions of variation after KDD, then we combine the result of PCA with the **DataFrame** to a final PCA representation of VALCRI as in table 5.21 .

	principal component 1	**principal component 2**
0	8.042236	29.623526
1	1.348689	2.602439
2	1.297932	2.224275
3	1.361436	2.351033
4	1.788163	1.948504

Table 5.20 VALCRI PCA components ONLY after KDD

This table 5.21 represent the PCA component concatenating with the VALCRI clustering dataset.

	principal component 1	**principal component 2**	**kmeans**
0	8.042236	29.623526	2
1	1.348689	2.602439	0
2	1.297932	2.224275	0
3	1.361436	2.351033	0
4	1.788163	1.948504	0

Table 5.21 The components of PCA Joined with clustering of VALCRI dataset

Note that the PCA of 2 component $pca.explained_variance_ratio$ between 0.14289058, 0.13561101. We also have used dimensionality reduction of 19 component represented as in table 5.22. In common, data projected it into a lower dimensional space uses the **LAPACK** implementation of the full **SVD** or a randomized truncated, and SVD by the method of *Halko* et al. 2009, depending on the shape of the input data and the number of components to extract. we used the following PCA module :

```
from sklearn.decomposition import PCA

pca = PCA(n_components=19)
pca.fit(ScaleNoDum)

PCA(copy=True, iterated_power='auto', n_components=19, random_state=None
    ,
svd_solver='auto', tol=0.0, whiten=False)
```

Listing 5.2 Dimensionality reduction Algorithms of 19 component

The table 5.22 show the result of the lower dimensional space uses the **LAPACK** implementation of the full **SVD** .

0.33535608	0.09047108	0.08393101	0.06178246
0.05991306	0.05278035	0.04775955	0.041053
0.03771201	0.03451236	0.03107427	0.02719704
0.02459588	0.01886537	0.01704837	0.01445389
0.01232751	0.00697453	0.00151566	

Table 5.22 Processed VALCRI dataset dimensionality reduction into **19** component

```
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler

features = data
# Separating out the features
x = df.loc[:, features].values

# Separating out the target
y = df.loc[:,['target']].values

# Standardizing the features
x = StandardScaler().fit_transform(x)


pca = PCA(n_components=19)
pca.fit(x_cols)


print(pca.explained_variance_ratio_)
plt.figure(figsize=(15,7))
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('number of components')
plt.ylabel('cumulative explained variance');
```

Listing 5.3 PCA Algorithms on the original dataset

5.6.3 Crime Dataset Features Selection

As we explained earlier this step is referred to by many names, data munching, data drangling and data pre-processing. The two broad categories of data wrangling are **feature selection** and **feature transformation**, we will discuss how we have applied those techniques.

Removing High Correlation in VALCRI dataset As we discussed in the previous section for outliers that we found an attributes in **VALCRI** dataset have undesirable outliers and which increases the error as in fig. 5.27g and these outliers are not the focus of our analysis, so we removed these outlier from our data set: so the graph in fig. 5.16 allowed us to write a function that we can identify the High Correlation then we created correlation matrix as in fig. 5.30 then we dropped the feature columns with correlation greater than 0.95 that store these values .

Then we examined these values $HOMC\_Code'$, $'HOOC\_Code'$, $'District\_Coded'$, and $'Beat\_Num\_Coded$ values we could see the result as in fig. 5.32 we as well have plot the correlated values as fig. 5.31 to see more distribution among them for final examination before dropping them. However there still more values needed to be investigated to include in the feature selection to extracting better knowledge form both dataset.

Removing High Correlation in VALCRI dataset At this point, both dataset been through very crucial tasks such as data quality, and preparation tasks that produced data frame clean, have non-missing , non-duplicated , non-categorical and non-string values that allowed us even to improve dataset further to have datasets without over-fitting and without outliers as result of several of feature transformation tasks

Removing features with low variance we used the removing features with low variance approach to remove feature that has low variance as selection method as in fig. 5.34 . As was expected, there seems to be a strong negative correlation between some values as in fig. 5.32 and we can also see visualisation as in fig. 5.33

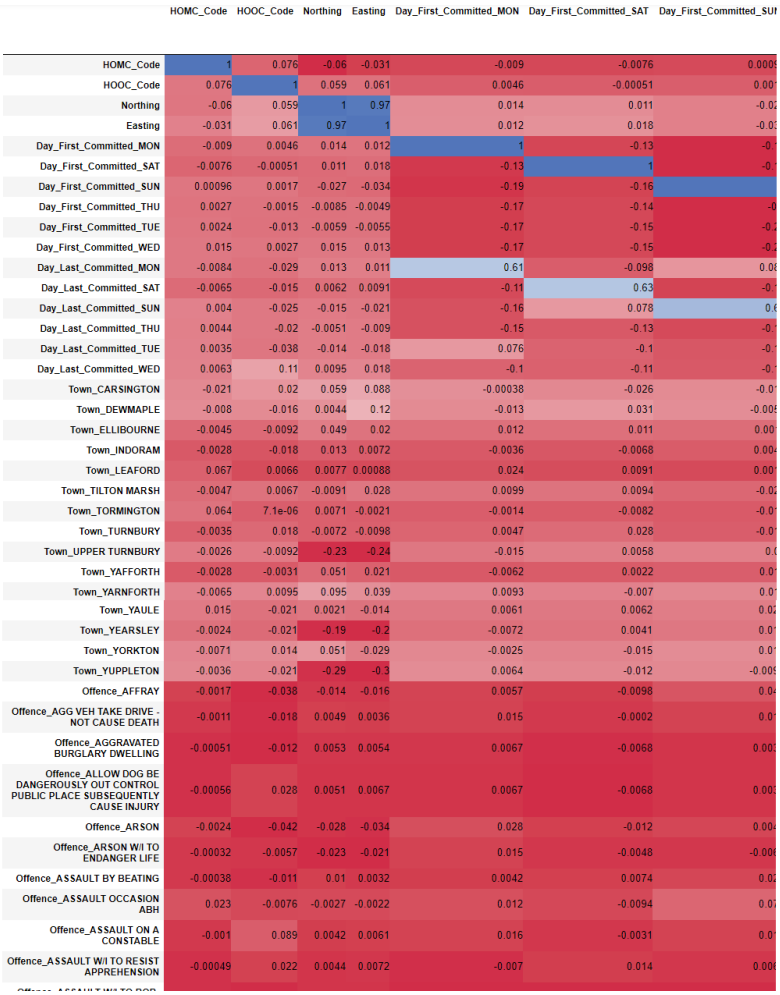
Then we created a scatter plot matrix using the *scatter_matrix* method that was available in "panda" [1] to analyse and access the data structures , then we removes all features whose variance does not meet some threshold. It removed all zero-variance features, and features

[1] "panda" library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language

that have the same value in all samples. we removed all features that are either one or zero (on or off) in more than 80% of the samples, and the variance of such variables is given by:

$$Var[x] = p(1-p)$$

$$sel = VarianceThreshold(threshold = (.8 * (1 - .8)))$$



	HOMC_Code	HOOC_Code	Northing	Easting	Day_First_Committed_MON	Day_First_Committed_SAT	Day_First_Committed_SUI
HOMC_Code	1	0.076	-0.06	-0.031	-0.009	-0.0076	0.000
HOOC_Code	0.076	1	0.059	0.061	0.0046	-0.00051	0.00
Northing	-0.06	0.059	1	0.97	0.014	0.011	-0.0
Easting	-0.031	0.061	0.97	1	0.012	0.018	-0.0
Day_First_Committed_MON	-0.009	0.0046	0.014	0.012	1	-0.13	-0.
Day_First_Committed_SAT	-0.0076	-0.00051	0.011	0.018	-0.13	1	-0.
Day_First_Committed_SUN	0.00096	0.0017	-0.027	-0.034	-0.19	-0.16	
Day_First_Committed_THU	0.0027	-0.0015	-0.0085	-0.0049	-0.17	-0.14	-0
Day_First_Committed_TUE	0.0024	-0.013	-0.0059	-0.0055	-0.17	-0.15	-0.
Day_First_Committed_WED	0.015	0.0027	0.015	0.013	-0.17	-0.15	-0.
Day_Last_Committed_MON	-0.0084	-0.029	0.013	0.011	0.61	-0.098	0.08
Day_Last_Committed_SAT	-0.0065	-0.015	0.0062	0.0091	-0.11	0.63	-0.
Day_Last_Committed_SUN	0.004	-0.025	-0.015	-0.021	-0.16	0.078	0.0
Day_Last_Committed_THU	0.0044	-0.02	-0.0051	-0.009	-0.15	-0.13	-0.
Day_Last_Committed_TUE	0.0035	-0.038	-0.014	-0.018	0.076	-0.1	-0.
Day_Last_Committed_WED	0.0063	0.11	0.0095	0.018	-0.1	-0.11	-0.
Town_CARSINGTON	-0.021	0.02	0.059	0.088	-0.00038	-0.026	-0.0
Town_DEWMAPLE	-0.008	-0.016	0.0044	0.12	-0.013	0.031	-0.00
Town_ELLIBOURNE	-0.0045	-0.0092	0.049	0.02	0.012	0.011	0.00
Town_INDORAM	-0.0028	-0.018	0.013	0.0072	-0.0036	-0.0068	0.00
Town_LEAFORD	0.067	0.0066	0.0077	0.00088	0.024	0.0091	0.00
Town_TILTON MARSH	-0.0047	0.0067	-0.0091	0.028	0.0099	0.0094	-0.0
Town_TORMINGTON	0.064	7.1e-06	0.0071	-0.0021	-0.0014	-0.0082	-0.0
Town_TURNBURY	-0.0035	0.018	-0.0072	-0.0098	0.0047	0.028	-0.0
Town_UPPER TURNBURY	-0.0026	-0.0092	-0.23	-0.24	-0.015	0.0058	0.0
Town_YAFFORTH	-0.0028	-0.0031	0.051	0.021	-0.0062	0.0022	0.0
Town_YARNFORTH	-0.0065	0.0095	0.095	0.039	0.0093	-0.007	0.0
Town_YAULE	0.015	-0.021	0.0021	-0.014	0.0061	0.0062	0.0
Town_YEARSLEY	-0.0024	-0.021	-0.19	-0.2	-0.0072	0.0041	0.0
Town_YORKTON	-0.0071	0.014	0.051	-0.029	-0.0025	-0.015	0.0
Town_YUPPLETON	-0.0036	-0.021	-0.29	-0.3	0.0064	-0.012	-0.00
Offence_AFFRAY	-0.0017	-0.038	-0.014	-0.016	0.0057	-0.0098	0.0
Offence_AGG VEH TAKE DRIVE - NOT CAUSE DEATH	-0.0011	-0.018	0.0049	0.0036	0.015	-0.0002	0.0
Offence_AGGRAVATED BURGLARY DWELLING	-0.00051	-0.012	0.0053	0.0054	0.0067	-0.0068	0.00
Offence_ALLOW DOG BE DANGEROUSLY OUT CONTROL PUBLIC PLACE SUBSEQUENTLY CAUSE INJURY	-0.00056	0.028	0.0051	0.0067	0.0067	-0.0068	0.00
Offence_ARSON	-0.0024	-0.042	-0.028	-0.034	0.028	-0.012	0.00
Offence_ARSON W/I TO ENDANGER LIFE	-0.00032	-0.0057	-0.023	-0.021	0.015	-0.0048	-0.00
Offence_ASSAULT BY BEATING	-0.00038	-0.011	0.01	0.0032	0.0042	0.0074	0.0
Offence_ASSAULT OCCASION ABH	0.023	-0.0076	-0.0027	-0.0022	0.012	-0.0094	0.0
Offence_ASSAULT ON A CONSTABLE	-0.001	0.089	0.0042	0.0061	0.016	-0.0031	0.0
Offence_ASSAULT W/I TO RESIST APPREHENSION	-0.00049	0.022	0.0044	0.0072	-0.007	0.014	0.00

Figure 5.30 correlation matrix

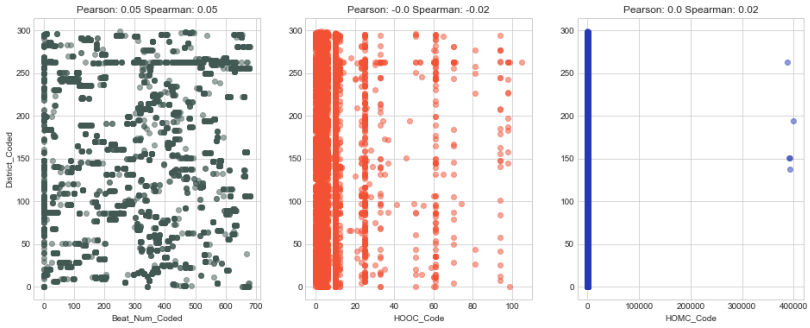


Figure 5.31 Dropped correlated values

	Northing	Easting	HOMC_Code	HOOC_Code	District_Coded	Beat_Num_Coded
Northing	1	-0.19	-0.066	0.00037	-0.087	0.037
Easting	-0.19	1	-0.092	0.038	-0.065	0.17
HOMC_Code	-0.066	-0.092	1	0.089	0.017	-0.087
HOOC_Code	0.00037	0.038	0.089	1	-0.025	-0.0033
District_Coded	-0.087	-0.065	0.017	-0.025	1	0.051
Beat_Num_Coded	0.037	0.17	-0.087	-0.0033	0.051	1

Figure 5.32 correlation relationship between quantities in VALCRI dataset

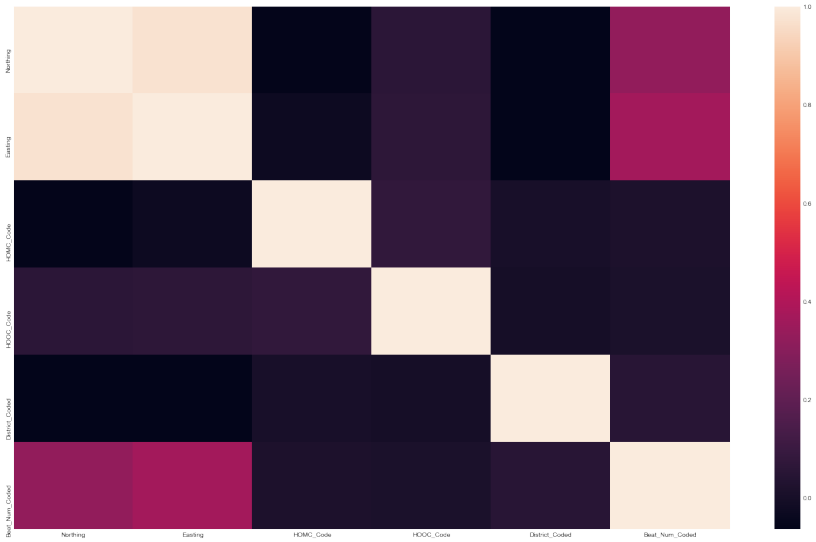


Figure 5.33 heatmap plot of correlation

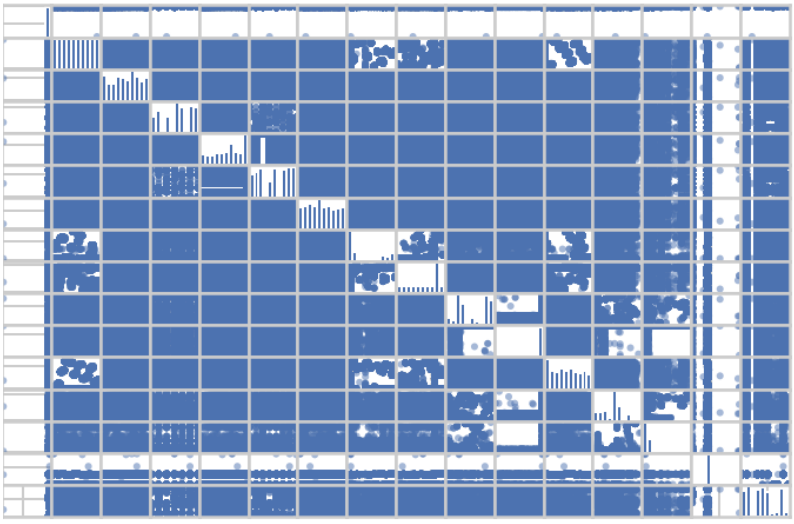


Figure 5.34 Removing features with low variance

Selection of relevant attributes We select features according to the K highest scores as in table 5.23, and it is a linear model for testing the individual effect of each of many regressors. This is a scoring function used in a feature selection procedure. This is done in 2 steps:

1. . The correlation between each regressor and the target is computed, that is in ,
2. . It is converted to an **F** score then to a **p-value**.

$$((X[:,i] - mean(X[:,i])) * (y - mean_y))/(std(X[:,i]) * std(y)). \tag{5.12}$$

```
# mutual_info_regression(X, y) -> Estimate mutual information for a
    continuous target variable.
selectors = []
# list of statistical functions
functions = [
f_classif,
mutual_info_classif,
f_regression
]

for f in functions:
sel = SelectKBest(f, k=2)
```

Score of:	f_ classif				
[	0.90545032	462.95684224	1.16536537	1.25053537	1.31882654
1.20222728	2.38467875	144.34844196	21.63406883	2.23198286	The
1.66605654	inf		6.35195037	2.59147342	43.53029523
2.90684633]					

Table 5.23 Selection of relevant attributes

[0.03852959	0.07821626	0.03924847	0.02345777	0.03834957	0.02258495
0.05979153	0.02761926	0.11281638	0.03182307	0.01098285	0.26777654
0.0571776	0.02855646	0.13683155	0.02623814]		

Table 5.24 Selection of SelectKBest attributes

```
12 sel.fit(X, Y)
13 selectors.append(sel)
14
15 pd.DataFrame(X).head()
```

Listing 5.4 SelectKBest relevant attributes Algorithms

Selection of SelectKBest attributes we used the select best method by the use of the $RandomForestClassifier()$ as in table 5.24 which uses averaging to improve the predictive accuracy and control over-fitting. table bellow shows the best features of the dataset

This section will conclude and demonstrate the result of the tasks been completed including identify the class distribution for feature selection and extraction which show in the table big different from the primary dataset and that helps extracting and selecting only relevant features for better Clustering representations of both crime dataset .

As well it is important to highlight the data also have different in the result of the Statistical calculation such as finding the Distribution of the dataset as in table 5.25.

The following table table 5.27 to compare ALL dataset with different result before and after scaling. Each Dataset has other name to the original dataset with different features, size , shape , and type. As well as each data can have different memory usage. However, this table for the purpose for illustration of the scale impact on the dataset before feature extraction, selection and before clustering.

Table 5.25 VALCRI Data **Non-Scaled** Statistic Distribution

	Street	District	Town	Post_Code	Offence	MO_Desc	Beat_Num
count	10000.000	10000.000	10000.000	10000.000	10000.00	10000.000	10000.00
mean	2138.229600	162.105900	5.374800	3177.579500	88.24330	1273.943100	312.14190
std	1238.499761	84.896316	5.503849	1578.809915	50.65164	329.436839	203.68371
min	0.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.00000
25%	1043.750000	87.000000	1.000000	1824.000000	44.00000	1407.000000	136.00000
50%	2076.000000	168.000000	2.000000	3661.500000	79.00000	1407.000000	306.00000

Table 5.26 VALCRI Data **Scaled** Statistic Distribution

	Street	District	Town	...	Offence	Beat_Num
count	1.000000e+04	1.000000e+04	1.000000e+04	...	1.000000e+04	1.000000e+04
mean	-9.874324e-17	9.243994e-17	3.984757e-16	...	-1.109879e-15	-1.094813e-15
std	1.000050e+00	1.000050e+00	1.000050e+00	...	1.000050e+00	1.000050e+00
min	-1.726554e+00	-1.909553e+00	-9.766018e-01	...	-1.742248e+00	-1.532560e+00
25%	-8.837582e-01	-8.847222e-01	-7.949017e-01	...	-8.735258e-01	-8.648247e-01
50%	-5.024846e-02	6.943051e-02	-6.132015e-01	...	-1.824968e-01	-3.015561e-02

Table 5.27 Data set Feature transformation comparison result

Name	Features	Numeric	Folat	Object	Sclaed	Dummies	Missing
0	data	19	2	3	14	False	No
1	IntNoDum	19	19	0	0	False	No
2	Missing0filling	19	2	3	14	False	No
3	NoDumScaled	18	18	0	0	True	No
4	dumNoscaled	185	183	2	0	False	Yes
5	dumScaled	185	0	185	0	True	Yes

```
print(results)
```

```
[[0, 0, 0, 1, 1, 2, 2, 0, 2, 2, 1, 2, 0, 0, 1, 2, 2, 0, 1, 0, 2, 0, 2, 0, 2, 0, 0, 2, 2, 0, 2, 2,
2, 2, 2, 2, 2, 1, 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 1, 0, 2, 2, 2, 0, 2, 2, 0, 1, 1, 2, 0, 2, 0, 2, 0,
2, 0, 2, 1, 0, 0, 2, 2, 2, 0, 1, 0, 2, 2, 2, 0, 0, 1, 1, 2, 0, 1, 2, 2, 0, 0, 2, 2, 2, 2, 1, 2, 2, 2,
0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 0, 2, 2, 2, 0, 1, 2, 0, 0, 2, 0, 0, 0, 1, 2, 2, 1, 0, 0, 0, 2, 2, 2,
1, 0, 1, 0, 0, 2, 1, 1, 2, 2, 0, 2, 0, 1, 2, 2, 2, 0, 2, 2, 0, 2, 0, 2, 0, 1, 2, 2, 0, 0, 1, 0, 2,
2, 0, 1, 2, 0, 2, 0, 1, 0, 0, 0, 0, 0, 0, 2, 1, 2, 1, 2, 1, 0, 1, 2, 0, 2, 0, 2, 1, 0, 1, 1, 0, 0,
0, 0, 0, 2, 0, 0, 0, 1, 2, 0, 2, 2, 2, 1, 2, 2, 0, 1, 0, 0, 0, 2, 2, 2, 1, 2, 2, 1, 1, 1, 0, 0, 0,
1, 1, 2, 2, 2, 1, 2, 2, 0, 1, 2, 1, 0, 0, 2, 0, 0, 2, 0, 1, 2, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 2, 1,
2, 1, 1, 0, 2, 1, 1, 2, 2, 0, 2, 1, 1, 0, 2, 0, 0, 2, 2, 0, 2, 0, 2, 0, 2, 1, 1, 0, 0, 1, 0, 0, 0,
1, 2, 0, 1, 0, 2, 2, 2, 1, 2, 2, 1, 0, 1, 0, 0, 2, 2, 1, 1, 0, 0, 1, 2, 0, 0, 0, 0, 2, 0, 2, 0, 0,
2, 2, 2, 1, 0, 2, 1, 2, 1, 0, 0, 1, 1, 1, 2, 1, 2, 2, 1, 2, 0, 2, 2, 2, 1, 1, 0, 0, 1, 1, 2, 2, 0,
0, 2, 0, 2, 2, 2, 2, 2, 2, 2, 1, 0, 2, 0, 0, 1, 0, 2, 2, 1, 0, 2, 0, 2, 2, 2, 2, 2, 2, 0, 0, 2, 0,
2, 0, 0, 0, 2, 2, 2, 2, 1, 2, 0, 2, 2, 0, 2, 1, 1, 2, 2, 0, 2, 0, 0, 2, 0, 1, 1, 0, 2, 1, 2, 1, 0,
2, 0, 1, 2, 1, 0, 1, 2, 2, 1, 0, 0, 1, 1, 2, 1, 1, 2, 0, 2, 1, 2, 2, 2, 0, 2, 2, 0, 1, 1, 2, 2, 0,
0, 1, 0, 2, 2, 0, 0, 2, 2, 0, 2, 0, 2, 0, 0, 1, 1, 1, 0, 0, 2, 0, 2, 0, 2, 2, 1, 1, 2, 0, 1, 2, 1,
1, 0, 0, 0, 1, 0, 2, 1, 0, 0, 0, 2, 2, 0, 2, 2, 2, 0, 0, 2, 2, 0, 1, 2, 2, 1, 0, 2, 0, 0, 0, 2, 0,
1, 0, 1, 0, 1, 0, 1, 2, 0, 1, 0, 2, 2, 2, 0, 1, 2, 1, 0, 2, 0, 2, 2, 0, 0, 0, 2, 1, 0, 2, 2, 1, 2,
1, 1, 2, 2, 0, 0, 1, 0, 2, 0, 2, 1, 2, 1, 1, 1, 2, 1, 0, 1, 0, 1, 2, 2, 0, 2, 2, 2, 2, 2, 0, 2, 2,
0, 1, 1, 1, 0, 2, 0, 2, 2, 0, 2, 2, 0, 0, 0, 1, 2, 0, 2, 2, 2, 2, 2, 1, 1, 1, 2, 2, 0, 2, 0, 0, 0,
```

Figure 5.35 Fitting all cluster Algorithms modules

5.6.4 Fitting all clusters modules

We train our module and fit all clustering modules to get the best feature selection , and we used K_means of 200 iteration, *agglom* algorithms that have linkage ward, and *spectral* affinity algorithms , as in fig. 5.35

```
from sklearn import cluster
k_clusters = 3
results = []
algorithms = {}

# K-Means Clustering
algorithms['kmeans'] = cluster.KMeans(n_clusters=k_clusters, n_init=200)
# Agglomerative Clustering
algorithms['agglom'] = cluster.AgglomerativeClustering(n_clusters=
    k_clusters, linkage="ward")
# Spectral Clustering
algorithms['spectral'] = cluster.SpectralClustering(n_clusters=
    k_clusters)
# Affinity Propagation
algorithms['affinity'] = cluster.AffinityPropagation(damping=0.6)
```

Listing 5.5 Fitting all cluster Algorithms modules

5.7 Clustering Algorithms on Crime Dataset

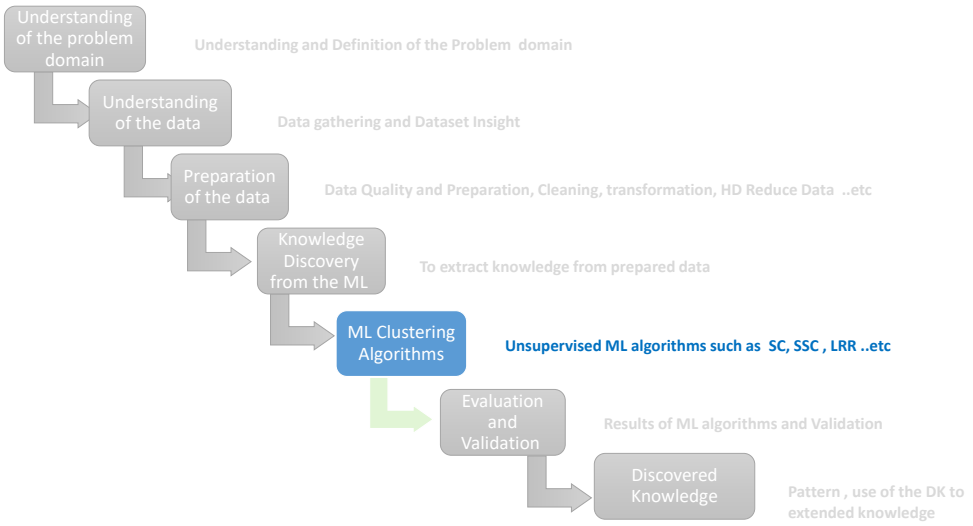


Figure 5.36 ML Clustering algorithms

The following results of unsupervised clustering algorithms applied to VALCRI data before and after KDD process , this result to compare the the initial dataset clustering algorithms result against dataset that been processed. There is already significant improvement as in following tables .

This section 5.7 show the result of the cluster of VALCRI dataset has been Transformed to Numeric using Dummies function but not scaled

This section 5.7 show the result of the cluster of VALCRI dataset that has been Numerically Transformed Data and Scaled.

This section 5.7 show the result of the cluster of VALCRI dataset that has been Scaled Data but not Numerically Transformed .

This section 5.7 show the result of the cluster of VALCRI dataset that that not Numerically Transformed nor Scaled .

Result of clustering all KDD dataset tasks This section 5.7 show the comparison result of clustering all VALCRI datasets at each tasks.

	Street	District	Post_Code	MO_Desc	Beat_Num	cluster
0	664	146	1887	-1	176	6
1	28	49	1491	-1	135	6
2	978	170	228	-1	194	6
3	2921	130	-1	-1	662	5
4	1934	35	4632	-1	404	2
5	1129	295	3632	-1	651	5
6	1472	166	4754	-1	408	2
7	4146	33	-1	-1	199	6
8	1791	256	3389	961	636	5

Table 5.28 Clustering Transformed Data to Numeric but not scaled

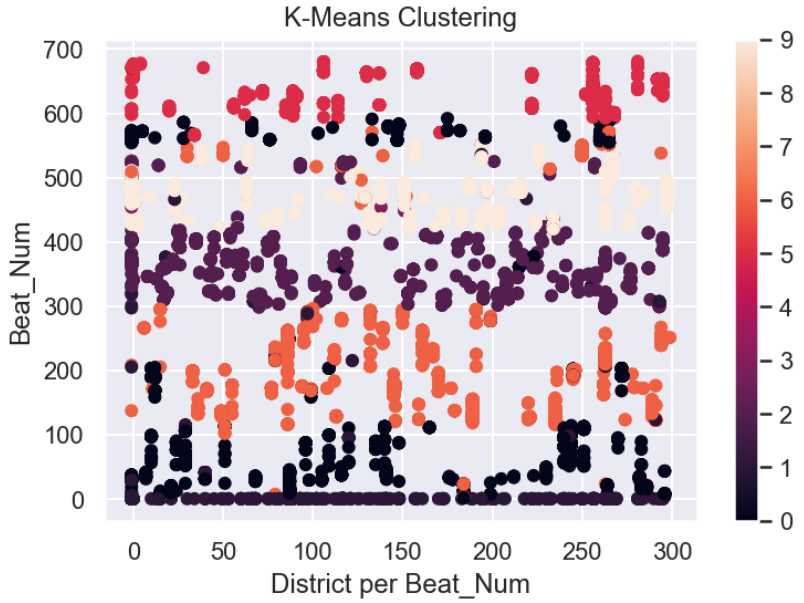


Figure 5.37 Clustering Transformed Data to Numeric but not scaled

	Street	District	Post_Code	MO_Desc	Beat_Num	cluster
0	27.0	49.0	1491.0	-1.0	135.0	7
1	977.0	170.0	228.0	-1.0	194.0	7
2	2920.0	130.0	-1.0	-1.0	662.0	2

Table 5.29 Clustering Numerically Transformed Data and Scaled

	Street	District	Post_Code	MO_Desc	Beat_Num	cluster
0	-1.704752	-1.332349	-1.068313	0.403912	-0.869735	4
1	-0.937657	0.104770	-1.868323	0.403912	-0.580055	4
2	0.632063	-0.378197	0.809144	0.403912	1.717740	7

Table 5.30 Clustering Scaled Data but not Numerically Transformed

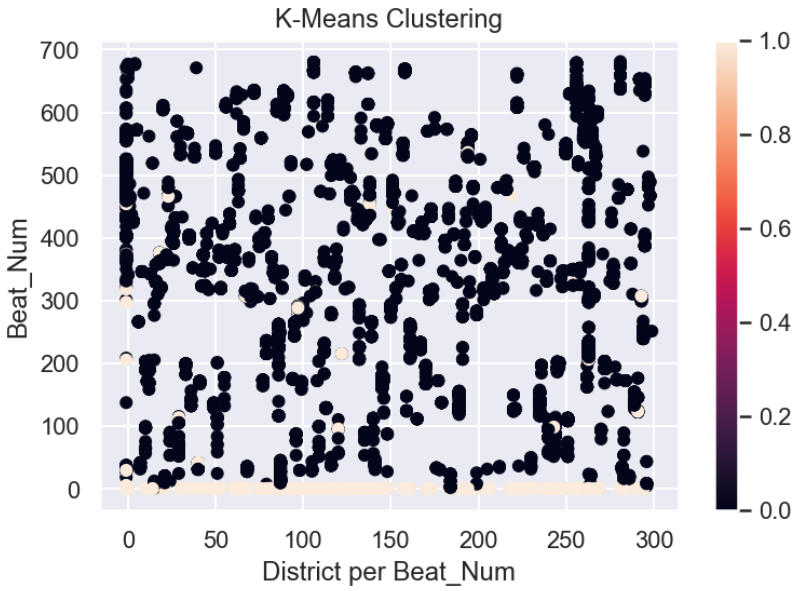


Figure 5.38 Clustering Numerically Transformed Data and Scaled

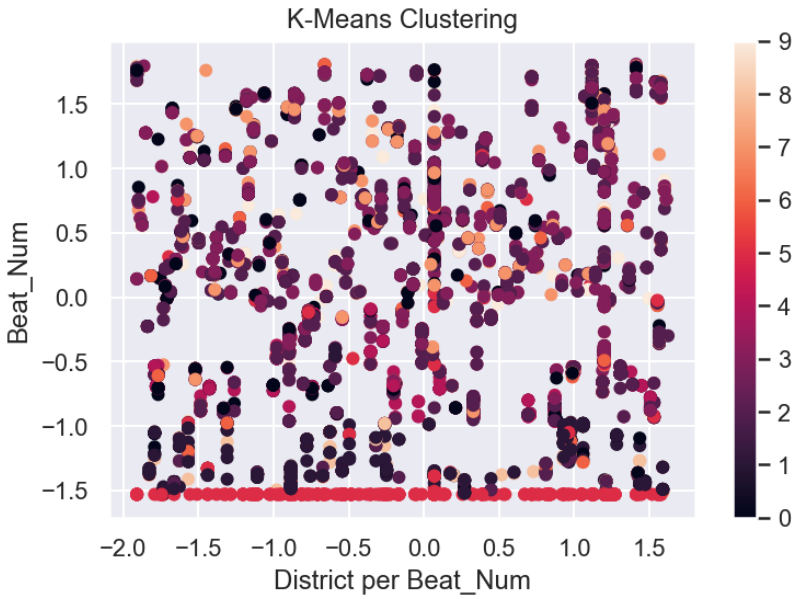


Figure 5.39 Clustering Scaled Data but not Numerically Transformed

	Street	District	Post_Code	MO_Desc	Beat_Num	cluster
0	**663**	**146**	**1887**	**1407**	**176**	3
1	**27**	**49**	**1491**	**1407**	**135**	2
2	**977**	**171**	**228**	**1407**	**194**	2

Table 5.31 Clustering Data that not Numerically Transformed nor Scaled

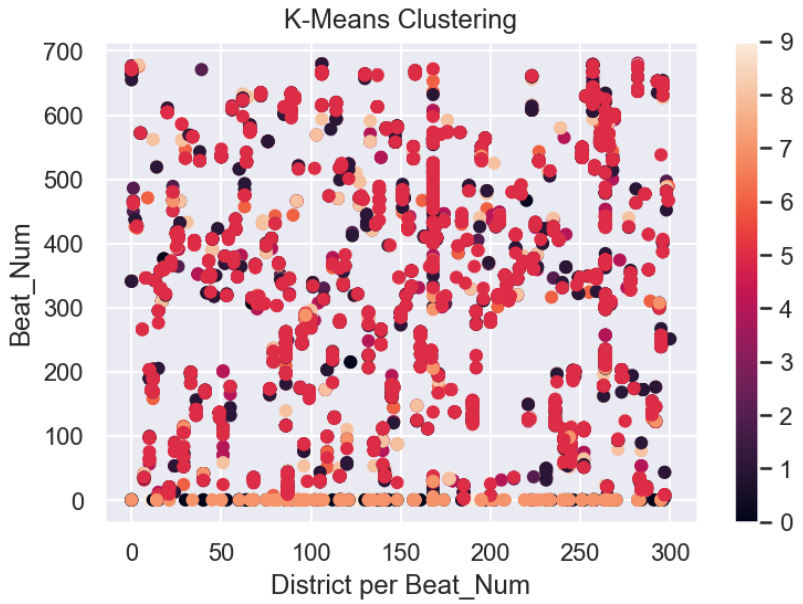


Figure 5.40 Clustering Data that not Numerically Transformed nor Scaled

	Feature	**dtypes**	**memory**	**clust_labels**
data	19	float64(2),int64(3),obj(14)	1.4 MB	N/A
ScaleDum	186	float64(185),int32(1)	14.2 MB	0.076
ScaleNoDum	20	float64(19),int32(1)	1.5 MB	0.098
DumNotScaleNum	225	int64(225)	17.2 MB	0.0874
DumNotScaleClean	186	float64(2),int32(1),int64(183)	14.2 MB	0.3506
NoDumNoScaleNum	20	int32(1),int64(19)	1.5 MB	0.334
NoDumNoScaleNotNum	19	float64(2),int64(3),obj(14)	1.4 MB	N/A

Table 5.32 ALL Clustering Data comparison Results

5.7.1 Robust PCA Data Reduction

Apply Robust PCA We apply RPCA using same as dataset output used for PCA in table 5.22 in previous section and we add the following:

RPCA = R _pca(X_scaled)

L, S = rpca.fit(max_iter=1000, iter_print=1)

iteration: 1, error: 63.29131964419159

To

iteration: 87, error: 0.009832057513205499

result show that

L is a Low Rank and S is Sparse Matrix of the data.

Change $max\_iter = 10000$, for better results

5.7.2 Spectral Clustering Algorithms (SC)

1. Data preparing from prepared dataset
2. Choosing our model
3. Fit model 80
4. Evaluation model 20
5. Hyperparameter
6. Prediction

Spectral clustering algorithms is subspace clustering methods, is well acknowledged to partitioning of a big data, and it come under the Graph-based clustering class. SC algorithms identified the data points as nodes in a weighted graph, and is to partition the nodes into several sets with the minimum sum of edge weights between each set. It is one of the two subspace clustering algorithms steps of the graph-based clustering methods that followed the subspace segmentation step which is to for constructing the affinity/similarity matrix , so the SC algorithms uses the affinity matrix to partitioning big dataset .

Benchmark Data Sets The data set from Package called '*mlbench*' which is a package with libraries that is included in "R" [2] , '**mlbench**' is a collection of artificial and real-world machine learning benchmark problems, including, e.g., *several data sets* from the **UCI** repository.

• **Details** of the data: Name **obj**, the summary can be found in table 5.33:

Table 5.33 obj data summary

	Length	Class	Mode
x	200	-none-	numeric
classes	100	factor	numeric

Results of creating data as obj **Example of the data**, we can see the head of the data as example in table 5.34:

Table 5.34 obj data Example

	x.1	x.2	classes
1	- 0.02253048	-0.4623878442	2
2	0.35187374	0.1379108955	1
3	0.16041228	0.8232393527	2
4	- 0.17414471	- 0.3863506083	2
5	0.30719916	0.1787081512	1
6	0.25101613	0.2070397047	1

inputs The inputs of the spirals problem are points on two *entangled* spirals. If $sd > 0$, as in fig. 5.41, ever the best wat to get sense of the data is to Visual it before the implementing the SC algorithms.

Usage:

specc :	$specc(my.data, centers = 2)$
Function:	$mlbench.spirals(n, cycles = 1, sd = 0)$
	$mlbench.spirals(100, 1, 0.025)$

Arguments for Function ***'mlbench.spirals'*** :

[2] R is a language and environment for statistical computing and graphics. R provides a wide variety of statistical such as **linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, ...). The topics that have used in this Benchmark Problems such as DNA, BreastCancer , Letter Recognition, bayesclass and Vote** .

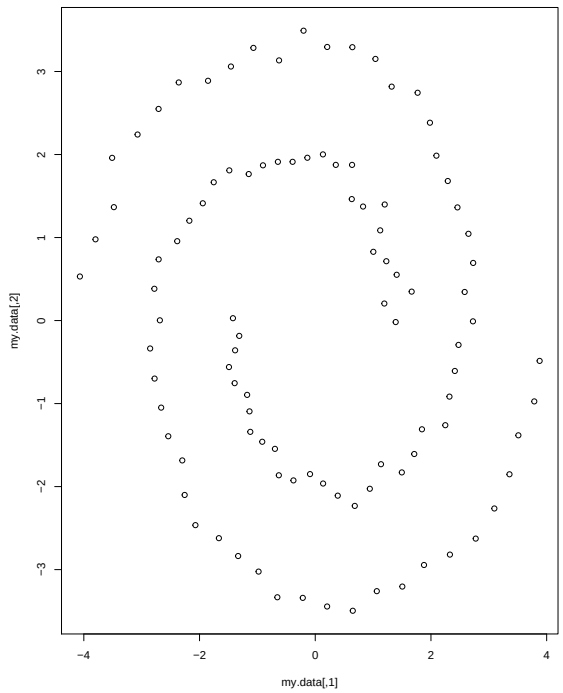


Figure 5.41 **obj** data plot before *SC algorithms* applied

n :	number of patterns to create
cycles :	the number of cycles each spiral makes
sd:	standard deviation of data points around the spirals
Value :	Returns an object of class "*mlbench.spirals*"

The Value of *mlbench.spirals* Returns an object of class with components: **x** input values **classes** factor vector of length n with target classes

Table 5.35 'Specc' Arguments

x:	the matrix of data to be clustered.
data:	an optional data frame containing the variables in the model.
centers :	a random set of rows in the eigenvectors matrix are chosen as the initial centers.
kernel : ***kernlab***	this function used in computing the affinity matrix. provides the most popular kernel functions

Arguments for Spectral Clustering '*Specc*' Note that the ***kernlab*** mentioned in table above has provides the most popular kernel functions which can be used by setting the kernel parameter to the following strings:

- *rbfdot* Radial Basis kernel function "**Gaussian**".
- *polydot* **Polynomial** kernel function.
- *vanilladot* **Linear** kernel function.
- *tanhdot* **Hyperbolic** tangent kernel function.
- *laplacedot* **Laplacian** kernel function.
- *besseldot* **Bessel** kernel function.
- *anovadot* **ANOVA RBF** kernel function.
- *splinedot* **Spline** kernel.
- *stringdot* **String** kernel.

The kernel parameter can also be set to a user defined function of class kernel by passing the function name as an argument.

Value: An S4 object of class *specc* which extends the class vector containing integers indicating the cluster to which each point is allocated.

The following points contain useful information:

- **centers** A matrix of cluster centers.
- **size** The number of point in each cluster.
- **withinss** The within-cluster sum of squares for each cluster .
- **kernelf** The kernel function used.

Table 5.36 my.data sample

	x.1	x.2
1	-0.09012193	-1.849551377
2	1.40749495	0.551643582
3	0.64164912	3.292957411
4	-0.69657885	-1.545402433
5	1.22879663	0.714832605
6	1.00406453	0.828158819

Extra step **x**: the matrix of data to be clustered, but the **data**: mentioned in table 5.34 has been scaled to $my.data < -4 * obj \$ x$ and then named under **my.data** , and in we can see how data changed :
Visual the implementing the SC algorithms :

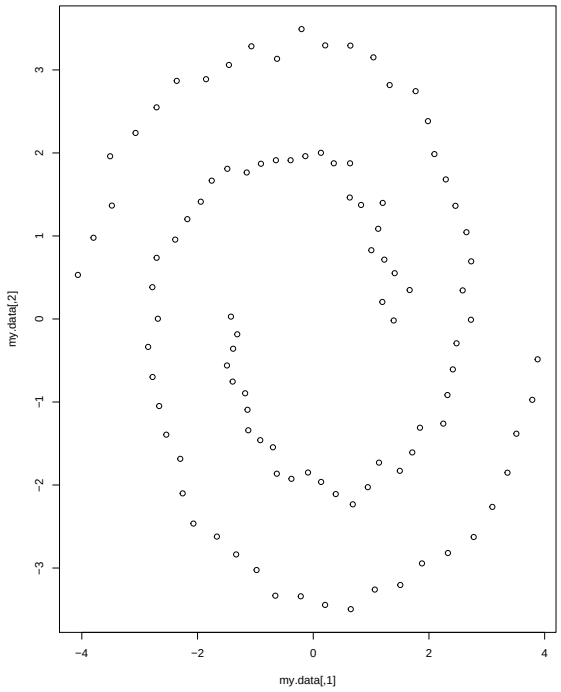


Figure 5.42 **my.data** from obj data

Implementation of SC

Usually any spectral algorithm is formed of the following three basic stages: pre-processing, spectral-representation and clustering as it is summarized in The steps bellow:

1. Project your data into $\mathscr{R}^n$
2. Define an *Affinity* matrix , using an Adjacency matrix or Gaussian Kernel K
3. Construct the Graph **Laplacian** from A by normalization
4. Solve an Eigenvalue problem
5. select **k** eigenvectors corresponding to the k lowest eigenvalues to define a *k-dimensional* subspace

6. Map each point to a lower-dimensional representation based on one or more eigenvectors

7. Clustering by assign points to two or more classes, based on the new representation.

To implement the SC by embedding the data into the **subspace** of the ***eigenvectors*** of an ***affinity matrix***.

Compute Similarity Matrix

- Spectral clustering needs a similarity or affinity 8 (x, y) measure determining how close points x and y are from each other.

- Let's denote the similarity matrix $\boldsymbol{S}$, as the matrix that at S_{ij} = 8 (x_i, x_j) gives the similarity between observations x_i and x_j .

- Common similarity measures are: + Euclidean distance: 8 (x_i, x_j) = $||x_i$ - $x_j||^2$ + Gaussian Kernel: s (x_i, x_j) = exp $(-\alpha||x_i$ - $x_j||^2)$

- compute **S** for my.data dataset using the *gaussian* kernel:

```
s <- function(x1, x2, alpha=1) {
exp(- alpha * norm(as.matrix(x1-x2), type="F"))
}

make.similarity <- function(my.data, similarity) {
N <- nrow(my.data)
S <- matrix(rep(NA,N^2), ncol=N)
for(i in 1:N) {
for(j in 1:N) {
S[i,j] <- similarity(my.data[i,], my.data[j,])
}
}
S
}

S <- make.similarity(my.data, s)
S[1:8,1:8]
```

Listing 5.6 compute **S** for my.data using the *gaussian* kernel

Results of the Affinity Matrix Representation on the my.data shown in the fig. 5.43

```
##           [,1]       [,2]      [,3]      [,4]       [,5]       [,6]
## [1,] 1.00000000 0.064179185 0.74290158 0.63193426 0.098831073 0.094897744
## [2,] 0.06417919 1.000000000 0.06938066 0.04276431 0.214229495 0.275123731
## [3,] 0.74290158 0.069380663 1.00000000 0.61054893 0.089569089 0.088641808
## [4,] 0.63193426 0.042764307 0.61054893 1.00000000 0.062517586 0.059982837
## [5,] 0.09883107 0.214229495 0.08956909 0.06251759 1.000000000 0.776556494
## [6,] 0.09489774 0.275123731 0.08864181 0.05998284 0.776556494 1.000000000
## [7,] 0.56549848 0.048335201 0.66577557 0.71959220 0.059731778 0.059016049
## [8,] 0.03355084 0.008796359 0.04342047 0.04426067 0.005091154 0.005548028
##           [,7]       [,8]
## [1,] 0.56549848 0.033550836
## [2,] 0.04833520 0.008796359
## [3,] 0.66577557 0.043420466
## [4,] 0.71959220 0.044260673
## [5,] 0.05973178 0.005091154
## [6,] 0.05901605 0.005548028
## [7,] 1.00000000 0.058059785
## [8,] 0.05805979 1.000000000
```

Figure 5.43 my.data Similarity Matrix

Affinity Matrix Representation

Then to to compute an affinity matrix **A** based on **S.A** ,and of positive values and be symmetric. to build a **representation** of a graph connecting by applying a k-nearest neighbour filter. However , to be symmetric, if A_{ij} is selected as a nearest neighbour, so will A_{ji}

```
make.affinity <- function(S, n.neighboors=2) {
N <- length(S[,1])

if (n.neighboors >= N) {  # fully connected
A <- S
} else {
A <- matrix(rep(0,N^2), ncol=N)
for(i in 1:N) { # for each line
# only connect to those points with larger similarity
best.similarities <- sort(S[i,], decreasing=TRUE)[1:n.neighboors]
for (s in best.similarities) {
j <- which(S[i,] == s)
A[i,j] <- S[i,j]
A[j,i] <- S[i,j] # to make an undirected graph, ie, the matrix becomes
    symmetric
}
}
}
A
}

A <- make.affinity(S, 3)  # use 3 neighboors (includes self)
```

```
A[1:8,1:8]
```

Listing 5.7 Affinity Matrix Representation computation

Results of the Affinity Matrix Representation on the **my.data** shown in the fig. 5.44, However, the an Affinity it is not the standard Euclidean metric but we here used Affinity Matrix to determines how close, or Similar, two points in space.

```
##          [,1] [,2]      [,3]      [,4]      [,5]      [,6]      [,7] [,8]
## [1,] 1.0000000    0 0.7429016 0.6319343 0.0000000 0.0000000 0.0000000    0
## [2,] 0.0000000    1 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000    0
## [3,] 0.7429016    0 1.0000000 0.0000000 0.0000000 0.0000000 0.6657756    0
## [4,] 0.6319343    0 0.0000000 1.0000000 0.0000000 0.0000000 0.7195922    0
## [5,] 0.0000000    0 0.0000000 0.0000000 1.0000000 0.7765565 0.0000000    0
## [6,] 0.0000000    0 0.0000000 0.0000000 0.7765565 1.0000000 0.0000000    0
## [7,] 0.0000000    0 0.6657756 0.7195922 0.0000000 0.0000000 1.0000000    0
## [8,] 0.0000000    0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000    1
```

Figure 5.44 Affinity Matrix Representation on the **my.data**

After computing the affinity matrix, replaced **clustering** is by a graph-partition problem, where connected graph components are interpreted as clusters. The graph are **partitioned** and that *edges* connecting different **clusters** should have low *weights*, and *edges* within the same cluster have high values. Spectral clustering is the technique that can construct this type of graph.

Degree Matrix D 'Diagonal Value'

The degree matrix **D** where each diagonal value is the degree of the respective vertex and all other positions are zero:

```
D <- diag(apply(A, 1, sum))
D[1:8,1:8]
```

Listing 5.8 Diagonal Value

Results of the Degree Matrix **D** 'Diagonal Value' on the **A** which is based on **A.S** shown in the fig. 5.45

Compute the Unnormalized Graph Laplacian

compute the **unnormalized** graph **Laplacian** ($U = D$ - A) and/or a normalized version **(L)**. However the **Laplacian** variants used :

```
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
## [1,] 2.374836 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
## [2,] 0.000000 2.597451 0.000000 0.000000 0.000000 0.000000 0.000000
## [3,] 0.000000 0.000000 2.408677 0.000000 0.000000 0.000000 0.000000
## [4,] 0.000000 0.000000 0.000000 2.351526 0.000000 0.000000 0.000000
## [5,] 0.000000 0.000000 0.000000 0.000000 2.523175 0.000000 0.000000
## [6,] 0.000000 0.000000 0.000000 0.000000 0.000000 2.519936 0.000000
## [7,] 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 3.170424
## [8,] 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
##          [,8]
## [1,] 0.000000
## [2,] 0.000000
## [3,] 0.000000
## [4,] 0.000000
## [5,] 0.000000
## [6,] 0.000000
## [7,] 0.000000
## [8,] 2.302241
```

Figure 5.45 Diagonal Value 'Degree Matrix'

- Simple Laplacian: I - $D^{-1}A$, where D^{-1} is the transition matrix. The spectral clustering yields groups of nodes such that the random walk seldom transitions from one group to another.
- Normalized Laplacian
- Generalized Laplacian:

we use the unnormalized **U**

```
U <- D - A
round(U[1:12,1:12],1)
```

Listing 5.9 The normalized Laplacian

Results of the The normalized **Laplacian** fig. 5.46 Assuming k clusters, the next step is to

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
##  [1,]  1.4  0.0 -0.7 -0.6  0.0  0.0  0.0  0.0  0.0   0.0   0.0   0.0
##  [2,]  0.0  1.6  0.0  0.0  0.0  0.0  0.0  0.0  0.0   0.0   0.0   0.0
##  [3,] -0.7  0.0  1.4  0.0  0.0  0.0 -0.7  0.0  0.0   0.0   0.0   0.0
##  [4,] -0.6  0.0  0.0  1.4  0.0  0.0 -0.7  0.0  0.0   0.0   0.0   0.0
##  [5,]  0.0  0.0  0.0  0.0  1.5 -0.8  0.0  0.0  0.0   0.0   0.0   0.0
##  [6,]  0.0  0.0  0.0  0.0 -0.8  1.5  0.0  0.0  0.0   0.0   0.0   0.0
##  [7,]  0.0  0.0 -0.7 -0.7  0.0  0.0  2.2  0.0  0.0  -0.8   0.0   0.0
##  [8,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.3  0.0   0.0   0.0   0.0
##  [9,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.5   0.0   0.0   0.0
## [10,]  0.0  0.0  0.0  0.0  0.0  0.0 -0.8  0.0  0.0   1.6  -0.8   0.0
## [11,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  -0.8   1.5  -0.8
## [12,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0   0.0  -0.8   1.5
```

Figure 5.46 normalized Laplacian

find the k smallest eigenvectors.

```
k   <- 2
evL <- eigen(U, symmetric=TRUE)
Z   <- evL$vectors[,(ncol(evL$vectors)-k+1):ncol(evL$vectors)]
```

Listing 5.10 The smallest eigenvectors

The i^{th} row of Z defines a transformation for observation x_i .
Results : Now we will see how they are Let's check that they are well-separated by plotting the Z to the *obj class* as in fig. 5.47

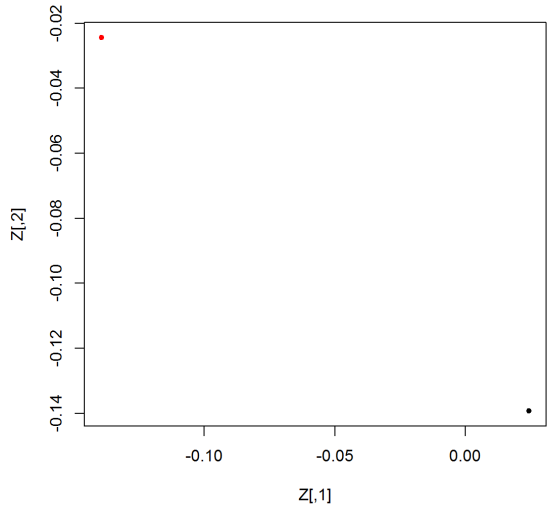


Figure 5.47 transformation for observation x_i defines by **Z**

Eigenvalue Spectrum for the Value of *'k'*

Note that , by now in this transformed space it becomes easy for a standard k-means clustering to find the appropriate clusters. However, at first was hard to determinate on know how much clusters there are, but the **eigenvalue spectrum** has a gap that give us the value of k:

```
signif(evL$values,2)
```

Listing 5.11 The smallest eigenvectors

Results of the smallest eigenvectors can be found in here fig. 5.49 :

```
##   [1] 3.3e+00 3.3e+00 3.0e+00 3.0e+00 2.9e+00 2.9e+00 2.8e+00 2.8e+00
##   [9] 2.7e+00 2.7e+00 2.7e+00 2.7e+00 2.6e+00 2.6e+00 2.6e+00 2.5e+00
##  [17] 2.5e+00 2.5e+00 2.5e+00 2.5e+00 2.4e+00 2.4e+00 2.4e+00 2.3e+00
##  [25] 2.3e+00 2.3e+00 2.3e+00 2.3e+00 2.2e+00 2.2e+00 2.2e+00 2.1e+00
##  [33] 2.1e+00 2.0e+00 2.0e+00 2.0e+00 2.0e+00 1.9e+00 1.9e+00 1.8e+00
##  [41] 1.8e+00 1.7e+00 1.7e+00 1.7e+00 1.6e+00 1.6e+00 1.6e+00 1.5e+00
##  [49] 1.5e+00 1.4e+00 1.4e+00 1.4e+00 1.3e+00 1.3e+00 1.2e+00 1.2e+00
##  [57] 1.1e+00 1.1e+00 1.1e+00 1.0e+00 9.7e-01 9.4e-01 8.8e-01 8.6e-01
##  [65] 7.9e-01 7.7e-01 7.1e-01 6.9e-01 6.2e-01 6.1e-01 5.5e-01 5.3e-01
##  [73] 4.7e-01 4.6e-01 4.1e-01 4.0e-01 3.4e-01 3.3e-01 2.8e-01 2.8e-01
##  [81] 2.3e-01 2.2e-01 1.8e-01 1.8e-01 1.4e-01 1.3e-01 1.0e-01 9.9e-02
##  [89] 7.0e-02 6.9e-02 4.4e-02 4.4e-02 2.5e-02 2.5e-02 1.1e-02 1.1e-02
##  [97] 2.8e-03 2.7e-03 2.8e-16 1.1e-16
```

Figure 5.48 The smallest eigenvectors **Results**

Results : can be seen of plotting the values that the eigenvalue spectrum has a gap as in fig. 5.49 , however, the This gap usually is hard to find. Choosing the optimal k is called "rounding" [3]

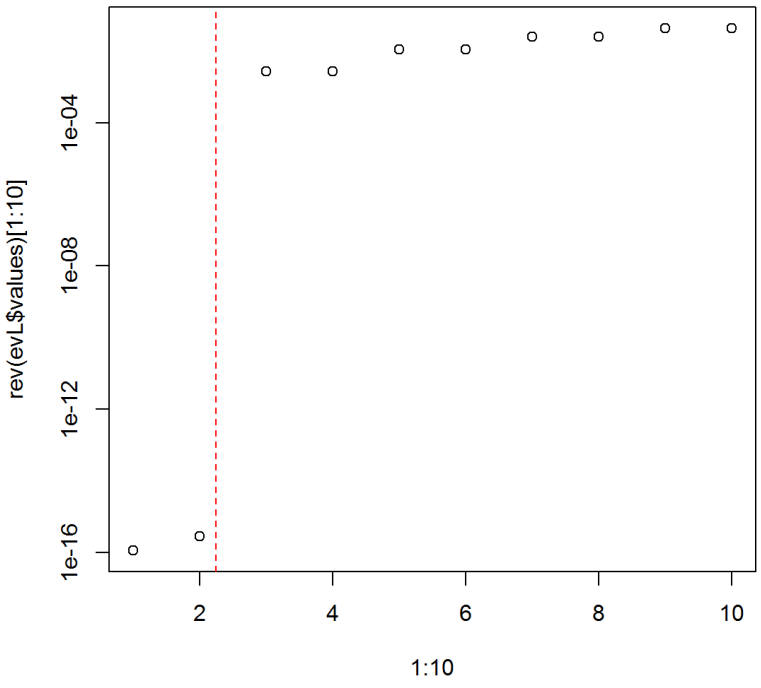


Figure 5.49 eigenvalue spectrum

[3]Rounding is an important method and step of **spectral clustering** , and the method is based on latent tree models. It can automatically select an appropriate number of **eigenvectors** to use, **determine** the number of clusters, and finally **assign** data points to clusters.

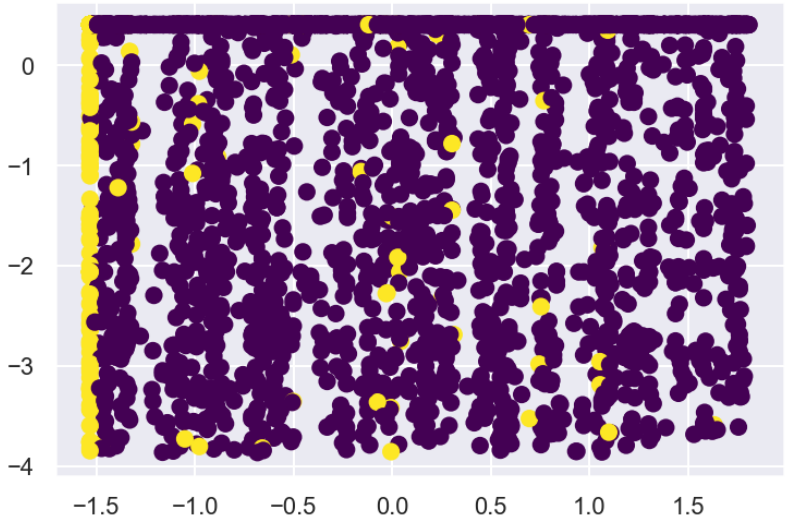


Figure 5.50 Spectral Clustering Algorithms (SC) to Criminal Data

5.7.3 Spectral Clustering Algorithms (SC) to Criminal Data

We applied clustering to a projection to the normalized *laplacian* as it very useful when the structure of the individual clusters is highly non-convex or more generally when a measure of the centre and spread of the cluster is not a suitable description of the complete cluster. This method used to find normalized graph cuts as in fig. 5.50.

5.8 Evaluation and Cross Validation

The process in this task is testing algorithms rapidly and discover whether or not there is structure in our problem for the algorithms to learn and which algorithms are effective and we solved by two tasks:

- Performance Measure: will give a score that is meaningful to domain.
- Create test and train datasets from the transformed data found in early tasks: to select a test set and a training set. An algorithm will be trained on the training dataset and will be evaluated against the test set.
- Cross validation : the performance measures are averaged across all folds to estimate the capability of the algorithm on the problem, it will be a $3-fold cross validation$ that involve training and testing a model 3 times as in figure then we will compare it with

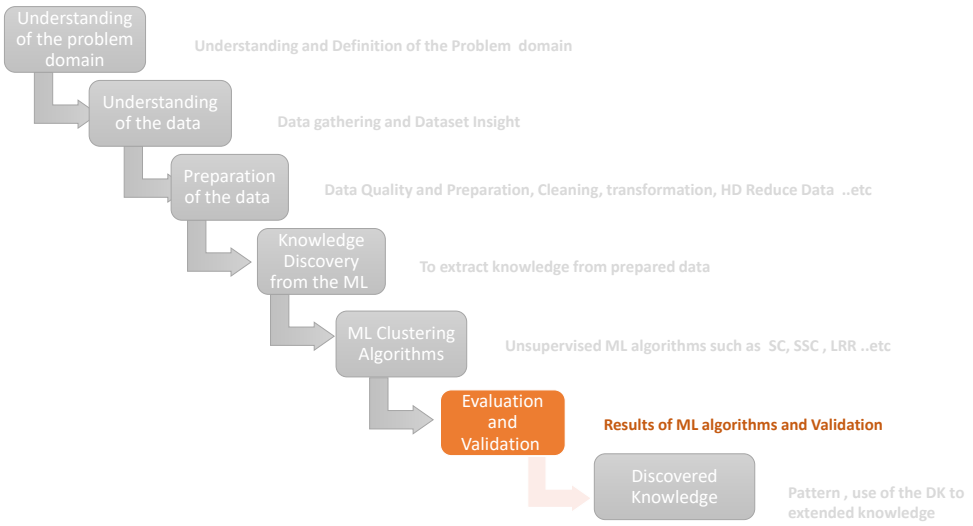


Figure 5.51 Evaluation and Validation

$5, 7, and 10 - foldcrossvalidation$ so the goal was to have a good balance between the size and representation of data in our train and test sets:

  - 1 : Train on folds 1+2 , test on fold 3
  - 2 : Train on folds 1+3 , test on fold 2
  - 3 : Train on folds 2+3 , test on fold 1

- Testing algorithms: We applied two type of performance evaluation for clustering techniques, first one is external evaluation in which we have previous information about data sets and second one is internal evaluation in which the evaluation is done with data set itself:
  - For external evaluation we used accuracy, and F-measure.
  - For internal performance measure we used validity indices silhouette index.

There are two important techniques that we used for machine learning algorithms evaluating to limit over-fitting: we used a resampling technique to estimate model accuracy. The most popular resampling technique is k-fold cross validation. After we have selected and tuned our machine learning algorithms on our dataset to evaluate the learned models to get a final

idea of how the models performed. Using cross validation was a valuable standard in applied machine learning for estimating model accuracy.

5.8.1 Clustering optimization

The first step in clustering analysis is to assess the method used for measuring similarities and the parameters used for partitioning.

Elbow Method

K-fold Cross Validation The purpose of k-fold cross validation as method to estimate the performance of an algorithms. Cross-validation systematically creates and evaluates multiple models on multiple subsets of the dataset. This provides result of performance measures. We calculated the mean of these measures to get an idea of how well the procedure performs on average. We calculated also the standard deviation of these measures to get an idea of how much the skill of the procedure is expected to vary in practice. This was also helpful for providing a more nuanced comparison of one procedure to choose which algorithm and best data preparation procedures to use. Also, this information is invaluable as you can use the mean and spread to give a confidence interval on the expected performance on a machine learning procedure in practice.

We used the elbow methods to optimizing a criterion, such as the within cluster sums of squares for elbow

$$\mu = 0, \sigma = 1 = 0, \sigma = 1 \tag{5.13}$$

where μ is the mean average and σ is the standard deviation from the mean; standard scores and called z scores of the samples are calculated as follows:

$$z = \frac{x - \mu}{\sigma} \tag{5.14}$$

Standardizing the features is not only important if we are comparing measurements that have different units, but it is also a general requirement for many machine learning algorithms, in the following fig. 5.52 we can see the Elbow method to select optimal clustering

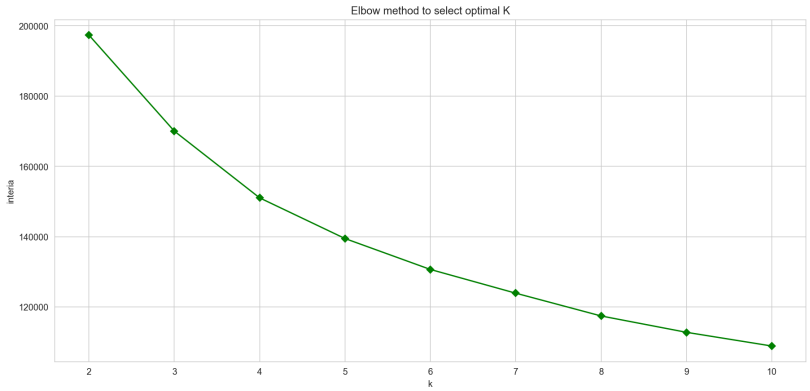


Figure 5.52 Elbow method to select optimal K

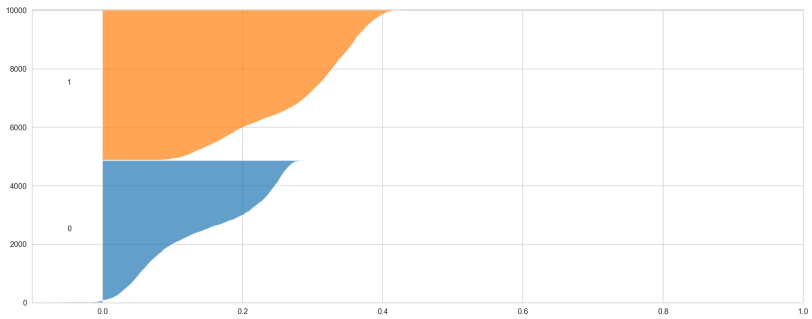


Figure 5.53 Silhouette Coefficient of all samples

Sillhoutte Method

we have analyse the clusters using *Sillhoutte* method to compute the mean Silhouette Coefficient of all samples as in fig. 5.53 , and the best value is 1 and the worst value is -1 as we can seen in . Values near 0 indicate overlapping clusters. Negative values generally indicate that a sample has been assigned to the wrong cluster, as a different cluster is more similar.

For	n_clusters	=	3	The	average	silhouette_score
For	n_clusters	=	4	The	average	silhouette_score
For	n_clusters	=	5	The	average	silhouette_score
For	n_clusters	=	6	The	average	silhouette_score
For	n_clusters	=	7	The	average	silhouette_score
For	n_clusters	=	8	The	average	silhouette_score

Table 5.37 Silhouette Coefficient of all samples

In addition to elbow, silhouette and gap statistic methods, there are more than thirty other indices and methods that have been published for identifying the optimal number of clusters.

In contrast to the metrics described above, this coefficient does not imply the knowledge about the true labels of the objects. It lets us estimate the quality of the clustering using only the initial, unlabeled sample and the clustering result. To start with, for each observation, the silhouette coefficient is computed. Let a be the mean of the distance between an object and other objects within one cluster and b be the mean distance from an object to an object from the nearest cluster (different from the one the object belongs to). Then the silhouette measure for this object is

$$s = \frac{b-a}{\max(a,b)}.$$

The silhouette of a sample is a mean value of silhouette values from this sample. Therefore, the silhouette distance shows to which extent the distance between the objects of the same class differ from the mean distance between the objects from different clusters. This coefficient takes values in the $[-1,1]$ range. Values close to -1 correspond to bad clustering results while values closer to 1 correspond to dense, well-defined clusters. Therefore, the higher the silhouette value is, the better the results from clustering.

With the help of silhouette, we can identify the optimal number of clusters k (if we don't know it already from the data) by taking the number of clusters that maximizes the silhouette coefficient.

VALCRI Feature ranking with recursive feature elimination RFE We set the goal of recursive feature elimination (RFE) and to select features by recursively considering smaller and smaller sets of features. In this section we will show how we selected features from processed dataset via **Recursive Feature Elimination (RFE)**, and the RFE by recursively removing attributes and building a model on those attributes that remain. It uses the model accuracy to identify which attributes and combination of attributes that contribute the most attribute. (RFE) is a feature selection method that fits a model and removes the weakest feature (or features) until the specified number of features is reached.

Approach : the goal of recursive feature elimination (RFE) is to select features by recursively considering smaller and smaller sets of features. In this step we have created new representation of the dataset that ready and Dropped the highly correlated features.

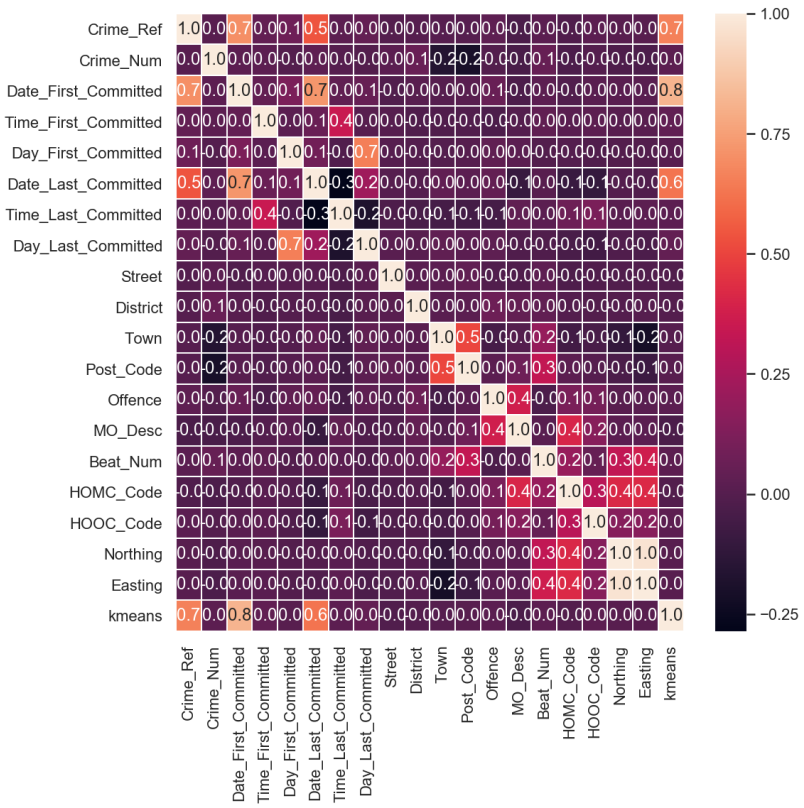


Figure 5.54 correlation matrix

We first create "**correlation matrix**" [4] as in fig. 5.54 then we Select upper triangle of correlation matrix as in table 5.38 , then we looked for the index of feature columns with correlation greater than 0.75% , Then, the least important features are pruned from current set of features.

[4]The "correlation matrix" is map to by visualizing the correlation matrix and to show different colours for each feature's correlations among the rest of the other feature. However, in fig. 5.54 the red colour with 0.0 means no correlations , and the lighter the colour, the larger the correlation magnitude, and it help to distinguishing positive from negative faster, as well as 0 from 1 which In addition to colour, we've added number as a parameter, and the number of each square corresponds to the magnitude of the correlation it represents .

	Crime_Ref	Crime_Num	..	target
Crime_Ref	NaN	0.027066	..	0.024628
Crime_Num	NaN	NaN	..	0.016986
Date1st_Committed	NaN	NaN	..	0.030211
Time1st_Committed	NaN	NaN	..	0.043536
Day1st_Committed	NaN	NaN	..	0.200983

Table 5.38 Processed VALCRI upper triangle of correlation matrix

```
# Create correlation matrix
corr_matrix = DumNotScaleNum.corr().abs()

# Select upper triangle of correlation matrix
upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).
    astype(np.bool))

# Find index of feature columns with correlation greater than 0.95
to_drop = [column for column in upper.columns if any(upper[column] >
    0.70)]
```

Listing 5.12 Recursive Feature Elimination (RFE) Algorithms

Next paragraph will show the cross-validation method used for the Feature Extraction RFE that used earlier applied on the VALCRI processed dataset.

Feature ranking with recursive feature elimination and cross-validated selection We have used estimator for learning estimator method to provides information about feature importance either through a $coef$ attribute and feature importances attribute. we set If greater than or equal to 1, then $step$ corresponds to the (integer) number of features to remove at each iteration. Then If within (0.0, 1.0), then $step$ corresponds to the percentage (rounded down) of features to remove at each iteration, and the last iteration removed fewer features in order to reach min features to select .

1. cross-validation We applied Logistic Regression CV **(aka logit, MaxEnt)** classifier. This approach for an estimator that has built-in cross-validation capabilities to automatically select the best hyper-parameters. The result of the $LogisticRegressionmethod$ = **0.0766**. In this case, the training algorithm uses the one-vs-rest scheme that if the $'multi_class'$ option is set to $'ovr'$, and uses the $cross-entropy$ loss if the $'multi_class'$ option is set to $'multinomial'$.

The advantage of using a cross-validation estimator over the canonical Estimator class along with grid search is that they can take advantage of warm-starting by reusing precomputed results in the previous steps of the cross-validation process. This generally leads to speed improvements. The liblinear solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. First, the estimator is trained on the initial set of features and the importance of each feature is obtained through a $coef_attribute$. Then, the least important features are pruned from current set of features. This procedure was recursively repeated on the set until the desired number of features was eventually reached , showing feature that ranked with [1] is important and extracted .

RFE Feature Ranking 1: which **Important Features Extracted:** [1, 'Crime_Ref'], [1, 'Crime_Num'], :., [1, 'HOOC_Code'] , [1, 'Northing'], [1, 'target']

2. best number of features by Feature ranking with RFE. We used the random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

```
class RandomForestClassifierWithCoef(RandomForestClassifier):
def fit(self, *args, **kwargs):
super(RandomForestClassifierWithCoef, self).fit(*args, **kwargs)
self.coef_ = self.feature_importances_

#iris = datasets.load_iris()
#x=pd.DataFrame(iris.data, columns=['var1','var2','var3', 'var4'])
#y=(pd.Series(iris.target, name='target')==2).astype(int)

x = NoDumNoScaleNum.drop(['District'], axis=1).fillna(0)
y = NoDumNoScaleNum['Beat_Num'].fillna(0)

rf = RandomForestClassifierWithCoef(n_estimators=20, min_samples_leaf=5,
    n_jobs=-1)
#rfecv = RFECV(estimator=rf, step=1, cv=2, scoring='roc_auc', verbose=2)
rfecv = RFECV(estimator=rf, step=1, cv=2, verbose=2)

selector=rfecv.fit(x, y)
```

Listing 5.13 Feature ranking with RFE Algorithms

The result of the selector ranking as in $array([8,4,11,15,10,13,5,7,2,12,16,1,6,9,3,14])$ showing the features ranking form 1 to 16 where 1 is most important and 16 least .

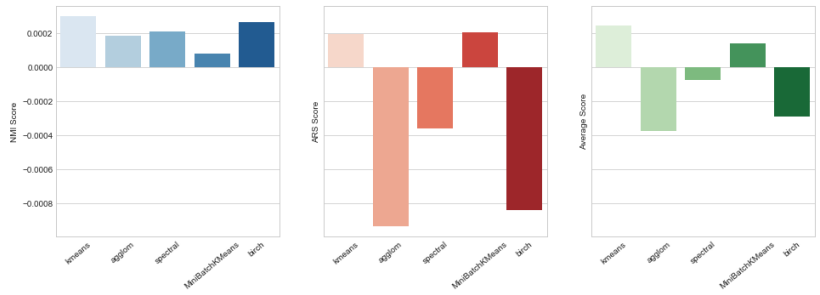


Figure 5.55 EVALUATION METRICS

5.8.2 Evaluation Metrics

The raw RI scored “adjusted for chance” into the ARI score as in fig. 5.55 using the following scheme:

$ARI = (RI - Expected_RI)/\ (max(RI)\ \text{-}\ Expected_RI)$The adjusted Rand index is thus ensured to have a value close to 0.0 for random labeling independently of the number of clusters and samples and exactly 1.0 when the clusterings are identical (up to a permutation).

Constraint and Limitation

Even the fact that SC has advantages such as using the affinity matrix and use local information to build a similarity matrix between pairs of points, still these methods are not good at handling corrupted data and sensitive to outliers. The constraint on the eigenvalue spectrum likewise existed, as Spectral clustering will only work on fairly "uniform datasets " [5]. some suggested that we must always using the Generalized Laplacian, Otherwise, there is no assurance that the full space can have eigen spectrums that line up correctly. Another Spectral clustering constraint that is used when K-means s works badly not because it is perfect module for big data , and because the clusters are not linearly separable in their original space.

[5] "Description" The Uniform Data System (UDS) is a core system of information appropriate for reviewing the operation and performance of health centres. UDS is a reporting requirement for Health Resources and Service Administration (HRSA) grantees, including community health centres, migrant health centers, health care for the homeless grantees, and public housing primary care grantees. The data are used to improve health center performance and operation and to identify trends over time. UDS data are compared with national data to review differences between the U.S population at large and those individuals and families who rely on the health care safety net for primary care [148]

5.9 Summary and Discussion

These strategies objective made the outcome of the processed data ready for advanced machine learning algorithms that improved how algorithms learnt, extraction of knowledge that implicitly stored or captured in the database.

The tasks of these strategies as fig. 5.1 has assist in creating new representation of the original data which is benefit form using the KDD Process for extracting useful Knowledge from volumes of Data identifying right features to link to relation in data, then it has helped for better view to uncover the implication and the problem of the original dataset. At the end will examine who has our strategies assist in discovering the pattern, knowledge that can be used as extended knowledge for future needs. The result of these methods that will be evaluated and discussed in details later at the end of this chapter to show the result and the contribution.

These strategies was to prepare, partition, process, transform the crime data before fitting it to the advanced subspace clustering as it made the outcome of the processed data ready for advanced machine learning algorithms that improved how algorithms learnt, extracted knowledge, and in overall how was the crime dataset before and after pre-pared . The result of these methods that evaluated in each section for each technique where the result shows.

The tasks used was advance how we gain more domain knowledge accurately and analyses the data that found unlabelled and the structure was unknown which mad it hard to discover relation, cluster or predict before been used. These tasks also has assisted in identifying best features link to relation in data , then helped in discovering the knowledge that has useful information that can be used to prevent or predict information, as well as used as extended knowledge for future needs.

Our work display use the evolution of engineering and many other disciplined that combined to be a a new strategy to deal with crime dataset or big data that have the **Outcome** of best data representations. The process helps to find the best way to set up better practise for big data problem.

The finding and experiments we have completed with the result of an depth looked at the related work mentioned earlier that led to discover that the traditional algorithms techniques are not very efficient enough to handle big dataset such as the criminal dataset, and most of these types of methods need to build an effective similarity matrix .

Chapter 6

Conclusion And Discussion

This chapter review the contributions that the thesis makes to support sense making through subspace clustering of high-dimensional data in criminal domain . It also discusses opportunities for future research on Applications To Criminal Data Analysis Based on Low Rank Sparse Representation Subspace Clustering

6.1 Review Of Contributions

The centre problem addresses in this thesis is how to design applications to criminal data analysis based on subspace clustering which has led us to implement of many subspace clustering. Our research has identified the most common problems and obstacles clustering high dimensional data specifically in criminal data. After discussion of all results from an evaluation of the algorithm applied that can bring forward the best practise of crime data analysis. In addition, we are interested on how to make the machine to learn in future to predict a new pattern using a new method to prevent possible crimes in the future. As in any applications there were limitations as explained in the previous section that defined where was the model more trusted and where does not.

6.1.1 PCA Segmented K-Means Clustering

PCA Segmented K-Means Clustering performs well when the data is disturbed by A random sample is a sequence of independent, identically distributed (IID) Guassian noise. PCA works well as long as the value of noise is not big enough, and It used by the feature transformation methods to help to reduce the data space to creates a new set of orthogonal variables that

keep the same information as the original set, but PCA dose not have as subspace clustering search methods and an evaluation criteria, and so the PCA used in feature transformation techniques does not help in this instance, since relative distances are preserved and the effects of the irrelevant dimension remain.

6.1.2 Crime data analysis

Based on our results using ranges of techniques we develop a new strategy that have the **Outcome** of best SC algorithms and data representations for our domain problem. The methods and the contribution to improve the way of criminal data investigating process and make data the outcome for machine learning algorithm the intuition was is to find the best way for best practise for big data problem for an algorithm. we demonstrated the methodology we used has improved the crime data investigating process for best outcome before applying any ML algorithms .

6.1.3 Subspace Clustering

Subspace clustering of high dimensional used to an interact with the input of the data based on the problems considered, and PCA has show a ineffective result, often even opposite to the possible solution. Subspace clustering advanced the exploited approach to the next level of subspace clustering that is LRSR for SC. The method been applied as partitioning dataset graph-based clustering class, and two steps to apply SC algorithms: subspace segmentation for constructing the affinity/similarity matrix and SC algorithms using the affinity matrix. However the implementation of SC not only to segmented and have graph-based class but also by embedding the data into **subspace** of ***eigenvectors*** of an ***affinity matrix***. Then to computing an affinity matrix **A** based on **S.A** , and must be made it of positive values and be symmetric. The optimization of statistical methods is usage of approaches such as the EM can leads to a local minimum, as statistical approaches normally use independent samples drawn from a mixture of probabilistic distributions to model data generation processes.

6.1.4 Low Rank Sparse Representation for Subspace Clustering

Based on the advantages have been found of subspace clustering, we have found an improved version which has better performance of these methods. Subspace segmentation and specular clustering is the 2 steps of the graph-based methods in the statistical categorise which has led us to the low rank subspace sparse representation (LRSR). LRSR was not only recovers

the low-rank subspaces but also get a relatively sparse segmentation with respect to disjoint subspaces or even overlapping subspaces. The experiments and the evaluate of these methods on our domain . The method in overall comes from the second categories of the global spectral clustering and it is similar to RPCA and segmented subspace , but it needs also to create and build an effective affinity or similarity matrix for high-dimensional data, then to use that matrices.

6.2 Future Work

However, our future work includes the investigation in a systematic way, as improved version of this method is needed, and can be further extended to other state-of-the-art methods, such as LRSR. After discussion of all results from an evaluation of the algorithm applied that can bring forward the models for predicting the crime. In addition, we are also interested more on how to make the machine to learn to predict a new pattern using a new method to prevent possible crimes in the future.

In coming weeks, after the feedbacks and the discussion i will take everything into consideration from my superiors, colleagues, and other researchers, and I will consider to seek more feedback to achieve the objectives and goals which this research represent.

6.3 Discussions

The research, related work, and the experiments we have completed through the use of the traditional clustering methods such as K-means, PCA segmented, and the low rank representation then we have discovered these are not very efficient enough to handle high dataset such as the criminal dataset, and most of these types of methods need to build an effective similarity matrix for high-dimensional data. When the data is high-dimensional, features are usually sparsely distributed and traditional methods such as computing distances directly in the original space are not reliable. Therefore the results of this research has lead us to find more suitable, recent, and advanced approach to be used in high dimensional space in order to cluster a large and complex data. Our research has identified the most common problems and obstacles clustering high dimensional data specifically in criminal data

This research, the experiments and analysis will lead us to the validation for the convergence of a subspace clustering framework called LRSR, which obtains the optimal solution based on both low rank representation (least number of dimensions) and sparse representation

(least number of support data). With low rank subspace recovery by detecting corruptions and limiting the self-expressive coefficient matrix to be sparse for disjoint subspaces, our proposed algorithm can reveal multiple subspace structures. The LRSR algorithm aims to recover the low-dimensional subspaces and to seek a low-rank and sparse representation for clustering.

However, future work includes the investigation of a systematic way is needed as improved version of this method, and can be further extended to other state-of-the-art methods. As well for the future work we need to find an alternate implementations. What other implementations of advanced SC algorithm are available? Perhaps an alternate implementation of the method can achieve better results on the same data. Each algorithm has a myriad of micro-decisions that must be made by the algorithm implementor. Some of these decisions may affect skill on any problem.

Bibliography

[1] Shyam Varan Nath. Crime pattern detection using data mining. In *International Conference on Web Intelligence and Intelligent Agent Technology*, WI-IATW '06, pages 41–44, Washington, DC, USA, 2006.

[2] Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: A review. *SIGKDD Explor. Newsl.*, 6(1):90–105, June 2004.

[3] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[4] PremNarayan Arya Singh Raghuwanshi. Comparison of k-means and modified k-mean algorithms for large data-set. *International Journal of Computing, Communications and Networking*, 2012.

[5] Robert L. Goldstone. The import and export of cognitive science. *Cognitive Science 30(6)*, 2006.

[6] R. Eshleman and H. Yang. A spatio-temporal sentiment analysis of twitter data and 311 civil complaints. In *2014 IEEE Fourth International Conference on Big Data and Cloud Computing*, pages 477–484, Dec 2014.

[7] E. Elhamifar, , and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 35(11):2765–2781, 2013.

[8] A. Rathor and M. Gyanchandani. A review at machine learning algorithms targeting big data challenges. In *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, pages 1–7, Dec 2017.

[9] Pat Langley et al. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall symposium on relevance*, volume 184, pages 245–271, 1994.

[10] F. Usama and U. Ramasamy. Data mining and knowledge discovery in databases. *ACM DL Digital Library*, 39(11):24–26, November 1996.

[11] Baolin Yi, Haiquan Qiao, Fan Yang, and Chenwei Xu. An improved initialization center algorithm for k-means clustering. In *Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on*, pages 1–4, Dec 2010.

[12] Q. Zhang and P. Yuan. Mixed spatial temporal characteristics based crime hot spots prediction. *IEEE*, 2016.

[13] Shiju Sathyadevan, Surya Gangadharan, et al. Crime analysis and prediction using data mining. In *Networks & Soft Computing (ICNSC), 2014 First International Conference on*, pages 406–412. IEEE, 2014.

[14] Hsinchun Chen, Wingyan Chung, Yi Qin, Michael Chau, Jennifer Jie Xu, Gang Wang, Rong Zheng, and Homa Atabakhsh. Crime data mining: an overview and case studies. In *Proceedings of the 2003 annual national conference on Digital government research*, pages 1–5. Digital Government Society of North America, 2003.

[15] U. Thongsatapornwatana. A survey of data mining techniques for analyzing crime patterns. In *2016 Second Asian Conference on Defence Technology (ACDT)*, pages 123–128, Jan 2016.

[16] B. Varma and V. Kumari. Mining popular crime patterns in spatial databases. *International Journal of Computer Applications*, 131(18):43–48, December 2015. Published by Foundation of Computer Science (FCS), NY, USA.

[17] Jacek Dajda, Roman Dębski, Aleksander Byrski, and Marek Kisiel-Dorohinicki. Component-based architecture for systems, services and data integration in support for criminal analysis. *Journal of Telecommunications and Information Technology*, 2012, 01 2012.

[18] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In *ACM sigmod record*, volume 29, pages 1–12. ACM, 2000.

[19] Robert A Hanneman and Mark Riddle. Introduction to social network methods, 2005.

[20] A Kirschner. Overview of common social network analysis software platforms. *San Fransisco, Monitor Group*, 2008.

[21] Rachel Boba Santos. *Crime Analysis with Crime Mapping*. SAGE Publications, Inc, 2017.

[22] UK Statistics Authority for Crime Statistics Advisory Committee. *Statistical and analytical guidance on crime and policing statistics For analysts working for Police and Crime Commissioners*. Home Office, 2018.

[23] Pro. Sir Charles Bean. Independent review of uk economic statistics: final report. Technical report, gov.uk, 2016.

[24] ChunWei Tsai, ChinFeng Lai, HanChieh Chao, and Athanasios V Vasilakos. Big data analytics: a survey. *Journal of Big Data*, 2(1):21, 2015.

[25] R Dasoriya. A review of big data analytics over cloud. *Consumer Electronics-Asia (ICCE-Asia), 2017 IEEE International Conference on*, 2017.

[26] Philip Russom et al. Big data analytics. *TDWI best practices report, fourth quarter*, 19(4):1–34, 2011.

[27] B. Gerhardt, K. Griffin, and R. Klemann. Unlocking value in the fragmented world of big data analytics. Technical report, Cisco Internet Business Solutions Group, June 2012.

[28] Intel. Steps it managers can take to move forward with big data analytics. Technical report, Intel IT Center, June 2012.

[29] R. D. Schneider. Big data and the use of hadoop. Technical report, John Wiley&Sons Canada, 2012.

[30] S. Singh and N. Singh. Big data analytics. In *International Conference on Communication, Information Computing Technology (ICCICT)*, pages 1–4, Oct 2012.

[31] F. Jianqing, H. Fang, and L. Han. Challenges of big data analysis. *National Science Review*, 1(2):293–314, 2014.

[32] Richard E Bellman and Stuart E Dreyfus. *Applied dynamic programming*, volume 2050. Princeton university press, 2015.

[33] Zena M. Hira and Duncan Fyfe Gillies. A review of feature selection and feature extraction methods applied on microarray data. In *Adv. Bioinformatics*, 2015.

[34] Mark Andrew Hall. Correlation-based feature selection for machine learning. *machine learning at niversity of Waikato Hamilton*, 1999.

[35] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, March 2003.

[36] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[37] H Peng, F Long, and C Ding. Feature selection based on mutual information criteria of max dependency max-relevance and mmin redundancy. *IEEE Trans. Pattern Anal. Mach. Intell*, 2005.

[38] K Balasubramanian B Sriperumbudur and G Lebanon. Ultrahigh dimensional feature screening via rkhs embeddings. *in Proc. 16th Int. Conf. Artif. Intell. Stat*, 2013.

[39] I Guyon, S Gunn, M Nikravesh, and LA Zadeh. Feature extraction: Foundations and applications (studies in fuzziness and soft computing) springer. *Secaucus, NJ, USA*, 2006.

[40] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Transactions on Automatic Control*, 26(1):17–32, February 1981.

[41] H Abdi and J. W. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):433–459, 2010.

[42] M Tabassum F Afrin, M AlAmin. Comparative performance of using pca with kmeans and fuzzy c means clustering for customer segmentation. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, 2015.

[43] Martin Thoma. A survey of semantic segmentation. *CoRR*, abs/1602.06541, 2016.

[44] Chad M. Holcomb and Robert R. Bitmead. Subspace identification with multiple data records: unlocking the archive. *Solar Turbines Inc., 4200 Ruffin Road, San Diego CA, USA.*, 2017.

[45] Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651 – 666, 2010. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR)19th International Conference in Pattern Recognition (ICPR).

[46] Matthew Shardlow. An analysis of feature selection techniques. *studentnet cs manchester Projects*, 2013.

[47] Andreas G. K. Janecek and Wilfried N. Gansterer. A comparison of classification accuracy achieved with wrappers, filters and pca. *University of Vienna, Research Lab Computational Technologies and Applications*, 2008.

[48] Mark Senn. with an example and how to select the right variables? *Machine learning mastery an introduction-to-feature-selection*, 2016. https://goo.gl/kOpBWV.

[49] Jiancheng Zhong, Jianxin Wang Wei Peng, Zhen Zhang, , and Min Li. A feature selection method for prediction essential protein. *TSINGHUA SCIENCE AND TECHNOLOGY*, 2015.

[50] Noelia S Marono, Amparo A Betanzos, and Maria T Sanroman. *Filter Methods for Feature Selection, A Comparative Study*, page 178. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[51] X Lin X.W. Chen. Big data deep learning:. *challenges and perspectives, IEEE Access*, April 2014.

[52] R.V. Singh and M.P.S. Bhatia. Data clustering with modified k-means algorithm. In *Recent Trends in Information Technology (ICRTIT), 2011 International Conference on*, pages 717–721, June 2011.

[53] Kari Torkkola and William M Campbell. Mutual information in learning feature transformations. In *ICML*, pages 1015–1022, 2000.

[54] T Joshua ., S Vinde, and L John C. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2016.

[55] Q Gan F Shen J. Zhao. An extended isomap for manifold topology learning with soinn landmarks. *International Conference on Pattern Recognition. IEEE Computer Society*, 2014.

[56] S T Roweis L K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science vol 290*, 2000.

[57] B Mukund and Schwartz Eric. The isomap algorithm and topological stability. *Science*, 295(5552):7–7, 2002.

[58] Y. Liang, F. Shen, J. Zhao, and Y. Yang. A fast manifold learning algorithm for dimensionality reduction. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 985–988, Nov 2016.

[59] Ian Jolliffe. *Principal Component Analysis*, chapter 1-14, page 488. Springer-Verlag New York, 2 edition, 2005.

[60] Lindsay I Smith. *Principal Components Analysis*. Cornell University, USA, February 26 2002.

[61] Jieping Ye, Ravi Janardan, and Qi Li. Two-dimensional linear discriminant analysis. In *Advances in neural information processing systems*, pages 1569–1576, 2005.

[62] Yizong C. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(8):790–799, August 1995.

[63] K. Fukunaga and L. D. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21:32–39, 1975.

[64] Pasi F and Olli V. Iterative shrinking method for clustering problems. *Pattern Recogn.*, 39(5):761–775, May 2006.

[65] Alex Rodriguez and Alessandro Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, 2014.

[66] R. Vidal, Yi Ma, and S. Sastry. Generalized principal component analysis (gpca). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1945, Dec 2005.

[67] Martin A. Fi and Robert C. B. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

[68] Lei Xu. Beyond pca learnings: From linear to nonlinear and from global representation to local representation. *,Proceedings of International Conference on Neural Information Processing, Invited Paper*, 1994.

[69] Xu Lei. Byy harmony learning, structural rpcl, and topological self-organizing on mixture models. *Neural networks the official journal of the International Neural Network Society*, 15, 2002.

[70] Lei Shi, Zhiyong Liu, Shikui Tu, and Lei Xu. Learning local factor analysis versus mixture of factor analyzers with automatic model selection. *Neurocomputing*, 139:3–14, 2014.

[71] Yan, Jingyu, Pollefeys, and Marc. *A General Framework for Motion Segmentation: Independent, Articulated, Rigid, Non-rigid, Degenerate and Non-degenerate*, pages 94–106. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[72] A. Goh and R. Vidal. Segmenting motions of different types by unsupervised manifold clustering. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6, June 2007.

[73] L. Zelnik-Manor and M. Irani. Degeneracies, dependencies and their implications in multi-body and multi-sequence factorizations. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–287–93 vol.2, June 2003.

[74] J. Shi and J. Malik. Normalized cuts and image segmentation, pattern analysis and machine intelligence. *IEEE Transactions on 22*, 2000.

[75] J. Shi S. X. Yu. Multiclass spectral clustering. *IEEE Conference on Cmputer Vision and Pattern Recognition (CVPR)*, 2003.

[76] Marina Meila and Jianbo Shi. A random walks view of spectral segmentation. In *MAchine Learning*, 2001.

[77] X. Wang and I. Davidson. Active spectral clustering. In *IEEE International Conference on Data Mining*, pages 561–568, Dec 2010.

[78] R. KANNAN. On clusterings: Good, bad and spectral. *Journal of the ACM*, 2004.

[79] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. *IEEE Trans on PAMI*, 1993.

[80] Cuimei Guo, Sheng Zheng, Yaocheng Xie, and Wei Hao. A survey on spectral clustering. In *World Automation Congress 2012*, pages 53–56, June 2012.

[81] Y. Deng, Q. Dai, R. Liu, Z. Zhang, and S. Hu. Low-rank structure learning via nonconvex heuristic recovery. *IEEE Transactions on Neural Networks and Learning Systems*, 24(3):383–396, March 2013.

[82] P. Zhou, Z. Lin, and C. Zhang. Integrated low-rank-based discriminative feature learning for recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 27(5):1080–1093, May 2016.

[83] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, Jan 2013.

[84] E. J. Candees, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 2011.

[85] R. Liu, Z. Lin, F. De la Torre, and Z. Su. Fixed-rank representation for unsupervised visual learning. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 598–605, June 2012.

[86] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. *International Conference on Machine Learning (ICML)*, 2010.

[87] Siming Wei and Zhouchen Lin. Analysis and improvement of low rank representation for subspace segmentation. *CoRR*, abs/1107.1561, 2011.

[88] Junbin Gao. Robust l1 principal component analysis and its bayesian variational inference. *Neural Computation*, 20(2):555–572, 2008.

[89] E. Kim, M. Lee, C. H. Choi, N. Kwak, and S. Oh. Efficient norm-based low-rank matrix approximations for large-scale problems using alternating rectified gradient method. *IEEE Transactions on Neural Networks and Learning Systems*, 26(2), Feb 2015.

[90] Emmanuel J., Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *J. ACM*, 58(3):11:1–11:37, June 2011.

[91] Y Ma Z Lin, M Chen. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *UIUC technical report UIULC510*, 2010.

[92] Zhouchen Lin, Arvind Ganesh, John Wright, LEQIN Wu, MINMING Chen, and Yi Ma. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. In *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, 2009.

[93] J F Cai, E J. Candes, and Z Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 99(4):1956–1982, 2010.

[94] B. Nasihatkon and R. Hartley. Graph connectivity in sparse subspace clustering. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 2137–2144, Washington, DC, USA, 2011. IEEE Computer Society.

[95] Eva L. Dyer, Aswin C. Sankaranarayanan, and Richard G. Baraniuk. Greedy feature selection for subspace clustering. *J. Mach. Learn. Res.*, 14(1):2487–2517, January 2013.

[96] M Soltanolkotabi and E J. Cands. A geometric analysis of subspace clustering with outliers. *Annals of Statistics*, 40,(4):2195–2238, 2012.

[97] M Soltanolkotabi and E J. Cands. Robust subspace clustering. *The Annals of Statistics*, 42(4):669–699, 2014.

[98] Gabe Cunningham. Non-flat regular polytopes and restrictions on chiral polytopes. *arXiv preprint arXiv:1706.00940*, 2017.

[99] Deborah Osborne and Susan Wernicke. *Introduction to crime analysis: Basic resources for criminal justice practice*. Psychology Press, 2003.

[100] S. V. Nath. Crime pattern detection using data mining. In *2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops*, pages 41–44, Dec 2006.

[101] Y. Arora and D. Goyal. Big data: A review of analytics methods techniques. In *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, pages 225–230, Dec 2016.

[102] Seref Sagiroglu and Duygu Sinanc. Big data: A review. In *Collaboration Technologies and Systems (CTS), 2013 International Conference on*, pages 42–47. IEEE, 2013.

[103] WE Howden and Suehee Pak. Problem domain, structural and logical abstractions in reverse engineering. In *Software Maintenance, 1992. Proceerdings., Conference on*, pages 214–224. IEEE, 1992.

[104] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.

[105] E. Begoli and J. Horey. Design principles for effective knowledge discovery from big data. In *Conference on Software Architecture and European Conference on Software Architecture*, pages 215–218, Aug 2012.

[106] Cristobal Romero and Sebastian Ventura. Educational data mining, a survey. *Expert systems with applications*, 33(1):135–146, 2007.

[107] Arabinda Nanda and Saroj Kumar Rout. Data miningand knowledge discovery in databases, an ai perspective. In *Proceedings of national Seminar on Future Trends in Data Mining*, 2010.

[108] Samir Farooqi. Data mining: An overview. *IASRI, Library Avenue, Pusa, New Delhi-110012*, 2009.

[109] Khalid Raza. Application of data mining in bioinformatics. *arXiv preprint arXiv:1205.1125*, 2012.

[110] Sholom M Weiss and Casimir A Kulikowski. *Computer systems that learn: classification and prediction methods from statistics, neural nets, machine learning, and expert systems*. Morgan Kaufmann Publishers Inc., 1991.

[111] Chidanand Apte and Se June Hong. Predicting equity returns from securities data with minimal rule generation. In *KDD Workshop*, pages 407–418, 1994.

[112] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[113] Sami Äyrämö and Tommi Kärkkäinen. Introduction to partitioning-based clustering methods with a robust example. *Reports of the Department of Mathematical Information Technology. Series C, Software engineering and computational intelligence*, 1, 2006.

[114] Karl-Heinrich Anders and Monika Sester. Parameter-free cluster detection in spatial databases and its application to typification. *International Archives of Photogrammetry and Remote Sensing*, 33(B4/1; PART 4):75–83, 2000.

[115] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. Knowledge discovery and data mining: Towards a unifying framework. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 82–88. AAAI Press, 1996.

[116] Shichao Zhang, Chengqi Zhang, and Qiang Yang. Data preparation for data mining. *Applied artificial intelligence*, 17(5-6):375–381, 2003.

[117] Ikbal Taleb, Rachida Dssouli, and Mohamed Serhani. Big data pre-processing a quality framework. In *Big Data Pre-Processing A Quality Framework*, 07 2015.

[118] Zena M Hira and Duncan F Gillies. A review of feature selection and feature extraction methods applied on microarray data. *Advances in bioinformatics*, 2015, 2015.

[119] Andrew Ng. Machine learning and ai via brain simulations, 2013.

[120] F. P. Shah and V. Patel. A review on feature selection and feature extraction for text classification. In *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 2264–2268, March 2016.

[121] Jason Brownlee. Discover feature engineering, how to engineer features and how to get good at it. *Machine Learning Process*, 2014.

[122] Ahmed Adeeb Jalal. Big data and intelligent software systems. *International Journal of Knowledge-based and Intelligent Engineering Systems*, 22(3):177–193, 2018.

[123] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19):2507–2517, 2007.

[124] L Ladha and T Deepa. Feature selection methods and algorithms. *International journal on computer science and engineering*, 3(5):1787–1797, 2011.

[125] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[126] Eric P Xing, Michael I Jordan, Richard M Karp, et al. Feature selection for high-dimensional genomic microarray data. In *ICML*, volume 1, pages 601–608. Citeseer, 2001.

[127] Avrim L Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1-2):245–271, 1997.

[128] Sanmay Das. Filters, wrappers and a boosting-based hybrid for feature selection. In *Icml*, volume 1, pages 74–81, 2001.

[129] Douglas Fisher, Ling Xu, and Nazih Zard. Ordering effects in clustering. In *Machine Learning Proceedings 1992*, pages 163–168. Elsevier, 1992.

[130] Roberto Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on neural networks*, 5(4):537–550, 1994.

[131] Brian V Bonnlander and Andreas S Weigend. Selecting input variables using mutual information and nonparametric density estimation. In *Proceedings of the 1994 International Symposium on Artificial Neural Networks*, pages 42–50. Citeseer, 1994.

[132] H Yang and John Moody. Feature selection based on joint mutual information. In *Proceedings of international ICSC symposium on advances in intelligent data analysis*, pages 22–25. Citeseer, 1999.

[133] Anil K Jain, Robert PW Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):4–37, 2000.

[134] Isabelle Guyon and André Elisseeff. An introduction to feature extraction. In *Feature extraction*, pages 1–25. Springer, 2006.

[135] Steve Golson. One-hot state machine design for fpgas. In *Proc. 3rd Annual PLD Design Conference and Exhibit*, volume 1, 1993.

[136] Blake Shaw and Tony Jebara. Structure preserving embedding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 937–944. ACM, 2009.

[137] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is nearest neighbor meaningful. In *International conference on database theory*, pages 217–235. Springer, 1999.

[138] T. Aljrees, D. Shi, D. Windridge, and W. Wong. Criminal pattern identification based on modified k-means clustering. In *2016 International Conference on Machine Learning and Cybernetics (ICMLC)*, volume 2, pages 799–806, July 2016.

[139] Maryam Fazel. *Matrix rank minimization with applications*. PhD thesis, PhD thesis, Stanford University, 2002.

[140] Zhouchen Lin, Risheng Liu, and Zhixun Su. Linearized alternating direction method with adaptive penalty for low-rank representation. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 612–620. Curran Associates, Inc., 2011.

[141] Ronen Basri and David W Jacobs. Lambertian reflectance and linear subspaces. *IEEE transactions on pattern analysis and machine intelligence*, 25(2):218–233, 2003.

[142] P. Favaro, R. Vidal, and A. Ravichandran. A closed form solution to robust subspace estimation and clustering. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1801–1807. IEEE Computer Society, 2011.

[143] Junfeng Yang, Wotao Yin, Yin Zhang, and Yilun Wang. A fast algorithm for edge-preserving variational multichannel image restoration. *SIAM Journal on Imaging Sciences*, 2(2):569–592, 2009.

[144] C. Aone B. Larsen. Fast and effective text mining using linear-time document clustering. *International Conference on Knowledge Discovery and Data Mining*, 1999.

[145] T. Kanade Tomasi. Shape and motion from image streams under orthography. *a factorization method, International Journal of Computer Vision*, 1992.

[146] L. G. Brown T. E. Boult. Factorization-based segmentation of motions,. *Proc. Workshop on Visual Motion, IEEE Computer Society,*, 1991.

[147] Zhouchen Lin, Risheng Liu, and Huan Li. Linearized alternating direction method with parallel splitting and adaptive penalty for separable convex programs in machine learning. *Machine Learning*, 99(2):287–325, 2015.

[148] healthypeople.gov. uniform data system. Technical report, Uniform Data System (UDS) Resources, June 18, 2018.

[149] Salvador García, Julián Luengo, and Francisco Herrera. *Data preprocessing in data mining*. Springer, 2015.

[150] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[151] Mohammed J Zaki, Wagner Meira Jr, and Wagner Meira. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.

Appendix A

Modified K-Mean Algorithm Algorithm: Modified approach (S, k) , S = $x1, x2, \cdots, xn$

$Input$: The number of clusters $k1(k1 > k)$ and a $dataset$

containing n objects $(Xij+)$

$Output$: A set of k clusters (Cij) that minimize the Cluster $-error\ criterion$

Algorithm steps

1. Compute the distance between each data point and all other data- points in the set D
2. Find the closest pair of data points from the set D and form a data-point set Am $(1 <= p <= k+1)$ which contains these two data- points, Delete these two data points from the set D
3. Find the data point in D that is closest to the data point set Ap, Add it to Ap and delete it from D
4. Repeat step 4 until the number of data points in Am reaches (n/k)
5. If $p < k+1$, then$p = p+1$, find another pair of data points from D between which the distance is the shortest, form another data-point set Ap and delete them from D, Go to step 4

Algorithm A

1. For each data-point set Am $(1 <= p <= k)$ find the arithmetic mean of the vectors of data points $Cp(1 <= p <= k)$ in Ap.
2. Select nearest object of each $Cp(1 <= p <= k)$ as initial centroid.

3. Compute the distance of each data-point di $(1 <= i <= n)$ to all the centroids $cj(1 <= j <= k+1)$ as$d(di, cj)$
4. For each data-point di, find the closest centroid $cj$and assign di to cluster j
5. Set Cluster Id $[i] = j$; $j : Id$ of the closest cluster
6. Set Nearest Dist $[i++]$ = d(di, cj)
7. For each cluster j (1<=j<=k) , recalculate the centroids
8. Repeat

Algorithm 1 pseudocode K-means algorithm

Input
For each data-point di
Compute its distance from the centroid of the present nearest cluster
If this distance is less than or equal to the present nearest distance, the data-point stays in the cluster
Else ;
For every centroid cj (1<=j<=k) Compute the distance (di, cj); Endfor Assign the data-point di to the cluster with the nearest centroid Cj
Set Cluster Id [i] =j
Set Nearest Dist [i] = d (di, cj);
Endfor
Output

Pseudo-code Algorithm

EVALUATION METRICS Pseudo-code Algorithm

Algorithm 2 pseudocode EVALUATION METRICS

Input
nmi_results = [] ars_results = []
y_true_val = list(Y)
for y_pred in results: nmi_results.append(normalized_mutual_info_score(y_true_val, y_pred)) ars_results.append(adjusted_rand_score(y_true_val, y_pred))
fig, (ax1, ax2, ax3) = plt.subplots(1, 3, sharey=True, figsize=(16, 5)) x = np.arange(len(Y))
avg = [sum(x) / 2 for x in zip(nmi_results, ars_results)]
xlabels = list(algorithms.keys())
sns.barplot(xlabels, nmi_results, palette='Blues', ax=ax1) sns.barplot(xlabels, ars_results, palette='Reds', ax=ax2) sns.barplot(xlabels, avg, palette='Greens', ax=ax3)
$ax1.set_ylabel('NMIScore')ax2.set_ylabel('ARSScore')ax3.set_ylabel('AverageScore')ax1.set_xticklabels(xlab$
for i, v in enumerate(zip(nmi_results, ars_results, avg)): ax1.text(i - 0.1, v[0] + 0.01, str(round(v[0], 2))) ax2.text(i - 0.1, v[1] + 0.01, str(round(v[1], 2))) ax3.text(i - 0.1, v[2] + 0.01, str(round(v[2], 2)))
plt.show()
Output

Appendix A

List of Publications

CRIMINAL PATTERN IDENTIFICATION BASED ON MODIFIED K-MEANS CLUSTERING, ©2016 IEEE, Proceedings of the 2016 International Conference on Machine Learning and Cybernetics, Jeju, South Korea, 10-13 July, 2016

Data mining methods like clustering enable police to get a clearer picture of criminal identification and prediction. Clustering algorithms will help to extracts hidden patterns to identify groups and their similarities. In this paper, a modified k-mean algorithm is proposed. The data point has been allocated to its suitable class or cluster more remarkably. The Modified k-mean algorithm reduces the complex nature of the numerical computation, thereby retaining the effectiveness of applying the k-mean algorithm. Firstly, the data are extracted from the communications and movements record after tracking the park visitors over three days. Then, the original data will be visualised in a graphical format to help make a decision about how many numbers to consider as the K cluster. Secondly, the modified k-means algorithm on the clusters initial centre sensitivity will be performed. This will link similar segments and determine the occurrence of each data point in every segment group rather than partitioning the entire space into various segments and calculating the occurrence of the data point in every segment. Thirdly, result checking and a comparison with the normal k-mean will be performed. The investigation will focus on the movement of people around the park where the crime occurred, and how people move and communicate in the park, how patterns change, and the movement of groups and individuals. The experiments show that the modified K-means algorithm leads to a better way of observing the data

to identify groups and their similarities and dissimilarities in the criminal dataset as a specific domain.

CRIMINAL DATA ANALYSIS BASED ON SUBSPACE CLUSTERING,(Draft) October, 2018

Clustering high-dimensional with large number of groups or overlapping spaces can make the basic machine learning task fail in finding effective pattern from dataset for further analysis. It become more difficult when the dataset have unknown features especially in crime dataset where the whole features in entire dimensional space need to be consider for each tasks such as data processing, dimensionally reduction, exploration or building prediction module. There are set of techniques used before the implementation of any data mining method called data engineers or preprocessing [149]. it is one of the most meaningful step for Knowledge Discovery [150], [151], and using the traditional steps for Crime data it will be imperfect, containing inconsistencies then wont be useful to identify patterns or capture the right group even with advance clustering algorithms. Moreover, hiding pattern in different subsets of other dimension and to discover or remove irrelevant noise only be found by advanced Machine Learning algorithms. This has motivated the use of specific crime data pre-process techniques accurately to identifies and select the correct features for both dimensional representation that would maximum variance to fewer dimensions to summarizes the original data well and then for clustering that will help to identify the meaningful patterns in our dataset. This paper will prove a better way to knowledge discovery process in dealing with specific domain such as criminal dataset and to detect the useful patterns. The approaches have exploited many measure for FE, FS to reduce the high dimensionality of the dataset that will have an accurate choice of features representation and dimensionality reduction technique for future clustering building the derived values for it is features that to be non-redundant and transformed into reduced set of features to determinate the subset of the initial features. hen we used advanced subspace clustering such as Subspace Segmentation (SS) to constructing the affinity/similarity matrix, then apply Spectral Clustering (SC) to characterizes data points as nodes in a weighted graph that obtains the optimal solution based on both low rank representation (LRR) and sparse representation (SR). The approaches for getting strong results only when selecting the suitable processes to analyse the application domain earlier of chosen any machine learning algorithms. The method have proven other way of handling high-dimensional before clustering and it is very necessary to use specific and non-tradition data processing techniques, so the data presented correctly to a subspace clustering, and throughout this paper we will discuss and comparing the results with previous approaches. Both UCI Machine Learning Repository, and crime database used to develop the methods and compared the best data reduction and clustering algorithms that fit for high dimensional crime dataset. We used high-level

programming language for statistics and Machine Learning such as Python, R, and Matlab in our experiments and to visualize the high-dimensional data.